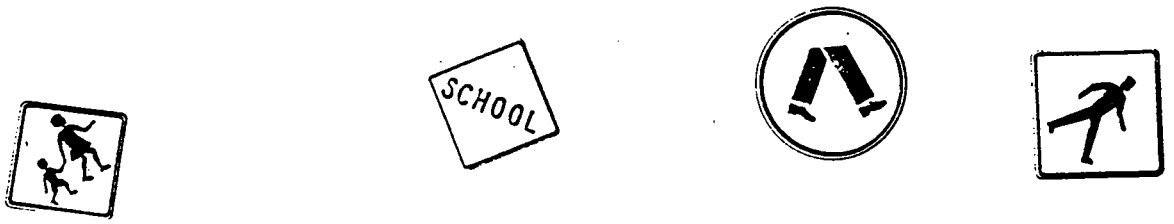
UNIVERSITY COLLEGE OF CENTRAL QUEENSLAND

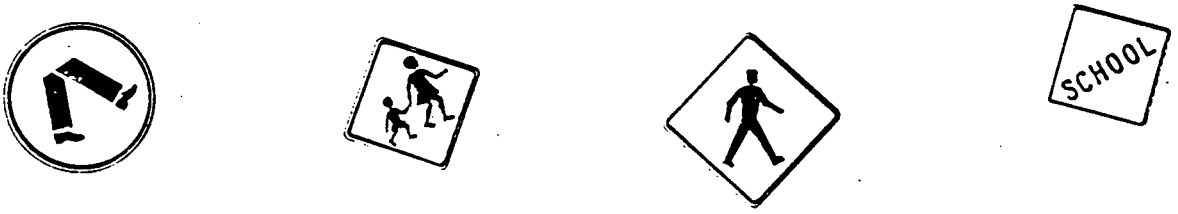
1990

Department of Civil Engineering

Subject: 61400 PROJECT (CIVIL)



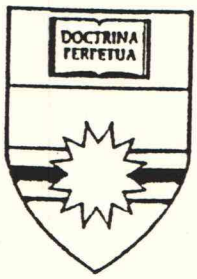
A STUDY OF PEDESTRIAN ACCIDENTS, BEHAVIOUR, AND SAFETY IN ROCKHAMPTON



Supervisor: Mr Akhtar Qizilbash

Due Date: December 31st 1990

by GERARD READ



UNIVERSITY COLLEGE OF CENTRAL QUEENSLAND

1990

Department of Civil Engineering

A STUDY OF PEDESTRIAN ACCIDENTS, BEHAVIOUR, AND SAFETY IN ROCKHAMPTON

ACKNOWLEDGEMENTS

I would like to thank the following people who helped in the production of this report.

* Mr Akhtar Qizilbash, project supervisor, for his guidance throughout the course of the project.
* Bevan Read, for his advice and assistance with the computer system.
* The Department of Transport, Armstrong St, Rockhampton, for supplying accident data for analysis.

TABLE OF CONTENTS

1 INTRODUCTION

Walking is the most basic means of transport. It offers predictable travel times, is free, and is considered by many as pleasant exercise. IT IS THE BACKBONE OF URBAN TRANSPORTATION PROVIDING A SYSTEM OF COLLECTION AND DISTRIBUTION WHICH ALLOWS THE CITY TO FUNCTION. In this day and age we are considered fortunate if we are located within walking distance of work, shopping, recreational or cultural opportunities. Unfortunately though, with walking comes accidents, injuries, and death as a result of the conflict the pedestrian undergoes with his surrounding environment.

Pedestrian safety is a significant road safety problem in Australia. In the past ten years over 5500 pedestrians have been killed on Australia's roads (almost 20% of the total road deaths). In addition to this nearly 3000 pedestrians are treated in hospital each year as a result of traffic accients.

This report deals with a study into pedestrian accidents, safety, and behaviour in the city of Rockhampton. The accidents studied were from 1985 - 1989. Due to "bureaucratical bungling", police traffic accident report forms were unable to be obtained for analysis, however the Department of Transport through the use of their computer traffic accident data base PHYLAK were able to give limited details of all pedestrian accidents occurring on the declared roads in Rockhampton. This numbered 27 for the five year period, and there were 82 overall reported in the city, a figure of 33%. Unfortunately personal details eg age, sex are not given in the PHYLAK data so it was impossible to analyse the relationship of these details to the actual accidents.

To gauge the public's reaction to pedestrian safety and behaviour a number of studies and surveys were conducted in the city. These involved obtaining a "pedestrian behaviour index" at various crossings throughout the city; determining the number of students who walk to school regularly; gauging difficulty associated with roundabouts; and the public's perception of safety on different types of crossings.

A computer data retrieval and analysis system was designed as a recommendation for the police to install, as a result of the confusion and turmoil experienced when attempting to gain the required information. With this program it would have been very simple indeed.

All these topics are discussed more extensively later in the report.

2 WHERE ROCKHAMPTON STANDS

2.1 Overview

The purpose of this chapter is to compare Rockhampton with ten other provincial Queensland cities, and overall on the national level, with regard to the number of pedestrian casualties and deaths incurred. The figures presented here are taken from the Bureau of Statistics, Brisbane publications (1985-1989) and statistics presented at the 15th ARRB Conference held at Darwin on Thursday 30 August 1990.

2.2 Nationally

Fig 2.1 shows the total number of casualties for Australia

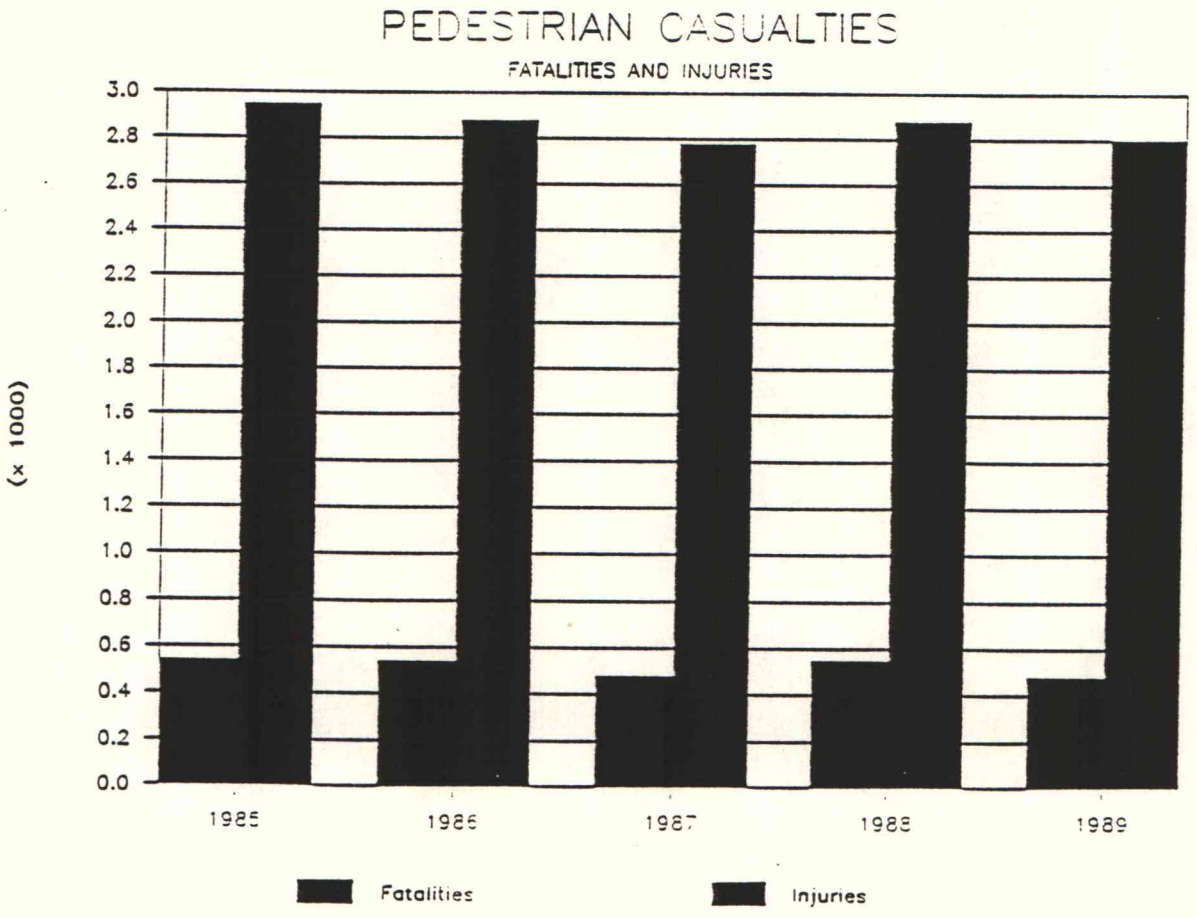


FIG 2.1 - Pedestrian Casualties in Australia (1985-89)

(Taken from the 15 ARRB Conference Paper)

for the years 1985-89. As can be seen from the figure, on average, 500 pedestrians are killed each year in Australia and around 2800 are injured. This represents almost 20% of the total road casualties nationally.

Rockhampton has contributed 5 deaths and 77 injuries resulting from pedestrian accidents during these years. This represents 1% of the national figure for fatalities and 2.75% for injuries. There have been 30 deaths and 854 injuries in total on the roads of Rockhampton in the 1985-89 period, making pedestrian deaths and injuries accountable for 16.7% and 9.0% respectively of the total road casualties. This puts Rockhampton slightly below the national average for fatalities and well below for injuries.

2.3 Within the State

To gauge Rockhampton's standing in the state with regard to pedestrian safety, an analysis was done covering ten other Queensland cities. These cities were Redcliffe, Ipswich, Toowoomba, Maryborough, Bundaberg, Gladstone, Mackay, Townsville Cairns, and Mt Isa. All information was gathered from the Bureau of Statistics. Figs 2.2 and 2.3 give the yearly breakdown of the number of injuries and deaths that occurred in each city. Rockhampton is ranked fourth behind Ipswich, Cairns, and Townsville in the number of fatalities and fifth behind the same cities plus Toowoomba in the number of injuries, for the five year period. This would indicate there is room for improvement in Rockhampton with regard to pedestrian safety in comparison with the rest of provincial Queensland. However the population of the cities needs to be also taken into account to acquire a more accurate assessment of the situation.

CITY	1985	1986	1987	1988	1989	TOTAL
Redcliffe	1	1	0	1	0	3
Ipswich	2	2	5	7	1	17
Toowoomba	1	0	0	1	2	4
Maryborough	0	0	0	0	1	1
Bundaberg	1	1	0	0	1	3
Gladstone	0	0	0	0	1	1
ROCKHAMPTON	1	1	1	0	2	5
Mackay	0	0	0	0	1	1
Townsville	2	2	2	1	1	8
Cairns	1	2	5	2	3	13
Mt Isa	0	2	0	1	0	3
TOTAL	9	11	13	13	13	59

FIG 2.2 - Pedestrian Deaths in Queensland Cities

CITY	1985	1986	1987	1988	1989	TOTAL
Redcliffe	16	6	13	10	8	53
Ipswich	29	25	22	27	18	121
Toowoomba	26	18	29	26	13	112
Maryborough	6	2	3	2	1	14
Bundaberg	10	11	9	8	6	44
Gladstone	7	2	4	4	7	24
ROCKHAMPTON	10	19	19	15	14	77
Mackay	6	10	12	7	7	42
Townsville	20	27	27	6	15	95
Cairns	14	18	19	24	21	96
Mt Isa	3	4	2	5	2	16
TOTAL	147	142	159	134	112	694

FIG 2.3 - Pedestrian Injuries in Queensland Cities

CITY	POPULATION	INJURIES	DEATHS
Redcliffe	47000	1.13	0.06
Ipswich	75000	1.61	0.23
Toowoomba	81000	1.38	0.05
Maryborough	23000	0.61	0.04
Bundaberg	33000	1.33	0.09
Gladstone	23000	1.04	0.04
ROCKHAMPTON	59000	1.31	0.08
Mackay	22000	1.91	0.05
Townsville	83000	1.14	0.10
Cairns	43000	2.23	0.30
Mt Isa	24000	0.67	0.13

FIG 2.4 - Pedestrian Injuries and Deaths per Capita (x 1000)

Fig 2.4 lists the injuries and deaths per head of population (x 1000) for the eleven cities including Rockhampton, for the period. Figs 2.5 and 2.6 present this information graphically for ease of comprehension. The horizontal line in the graphs is the Rockhampton figure.

These figures indicate that Rockhampton is the "median" centre for pedestrian casualties in the provincial areas of the state. In both cases there are five cities with higher rates, and five cities with lower rates. Rockhampton has a rate of 1.31 pedestrian injuries and 0.08 pedestrian deaths per thousand people. Cairns and Ipswich are by far the worst centres for overall pedestrian casualty rates, whilst Mackay has a relatively high injury rate but one of the lowest fatality rates. Bundaberg is above Rockhampton on both counts and Maryborough has the best record of all the cities, closely followed by Gladstone.

Although Rockhampton is sitting roughly average with reference to casualties, it means there is still room for an improvement to lower these rates.

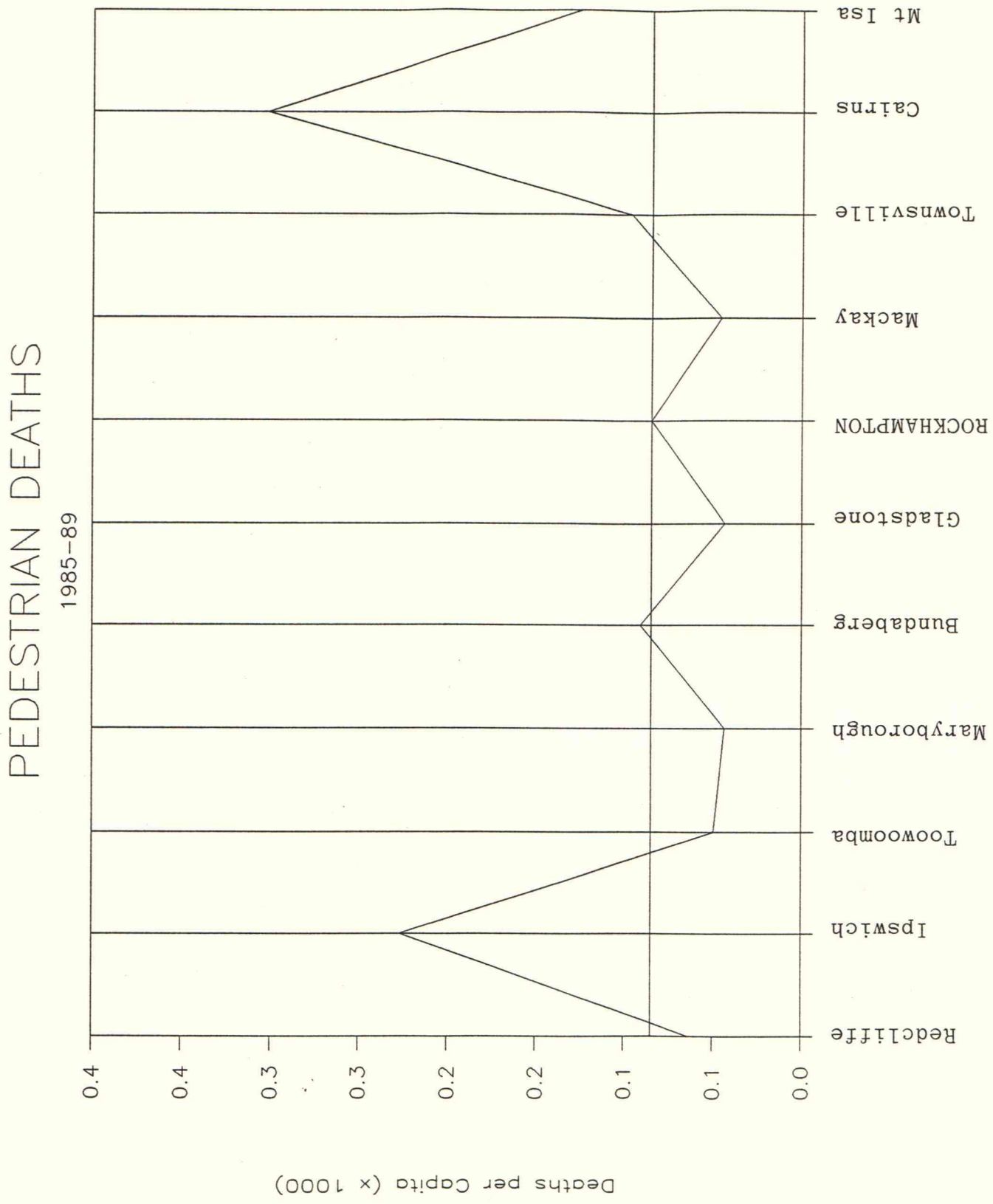


FIG 2.5 - Pedestrian Deaths per Capita

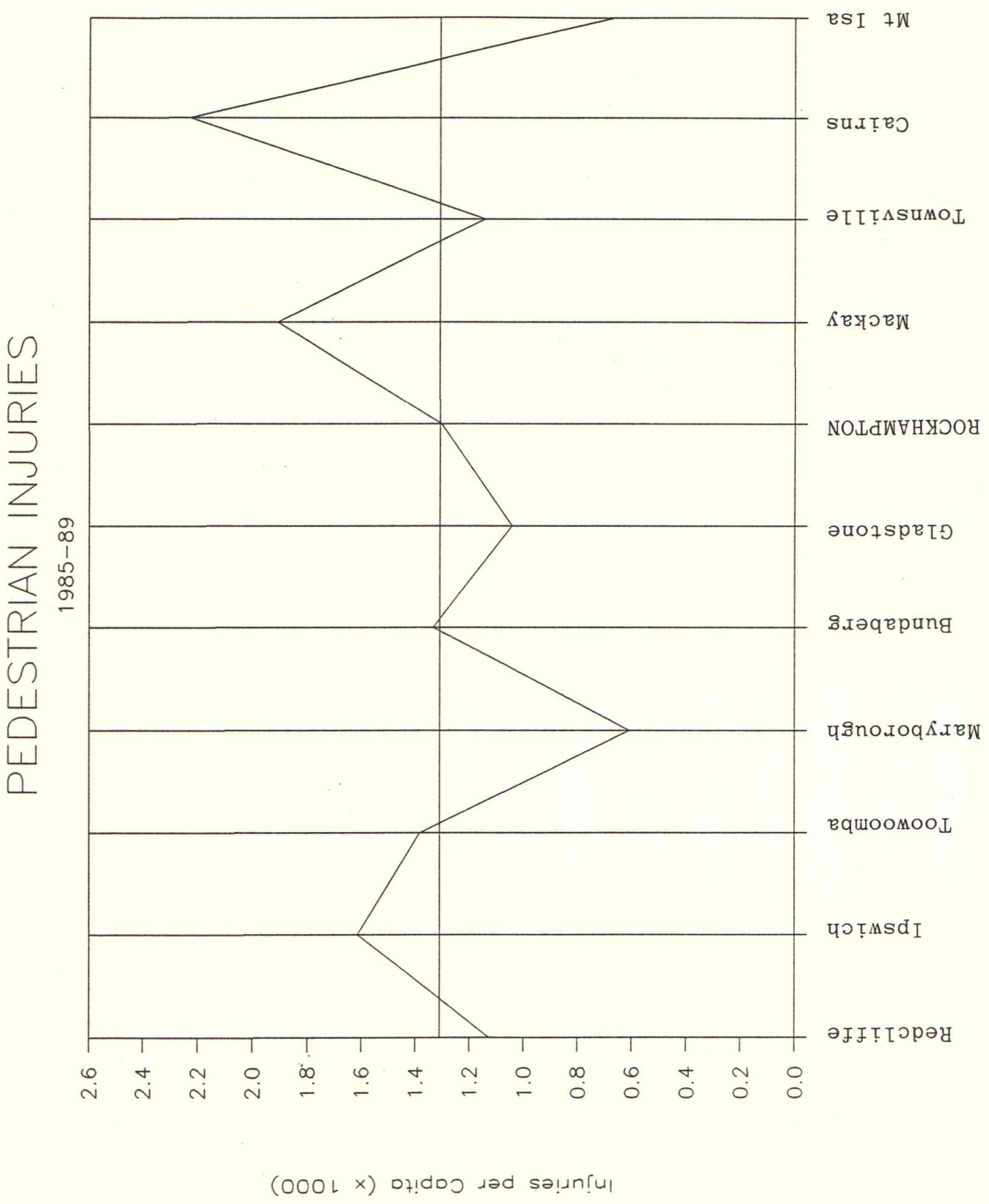


FIG 2.6 - Pedestrian Injuries per Capita

3 ACCIDENT CHARACTERISTICS

3.1 Local Conditions

Figure 3.1 summarises the local conditions at the site and time of each accident that took place. The road conditions, light conditions, and atmospheric conditions are the factors that are examined here. The information was taken from the Department of Transport data.

LOCAL CONDITIONS	NUMBER OF ACCIDENTS	% OF TOTAL ACCIDENTS
ROAD CONDITIONS		
Sealed Dry	23	85.2
Sealed Wet	4	14.8
Unsealed Dry	0	0.0
Unsealed Wet	0	0.0
LIGHT CONDITIONS		
Daylight	14	51.9
Dawn/Dusk	1	3.7
Dark	1	3.7
Lit	11	40.7
ATMOSPHERIC CONDITIONS		
Clear	22	81.5
Rain	4	14.8
Smoke/Dust	1	3.7
Fog	0	0.0

FIG 3.1 - Local Conditions of each Accident

The above table indicates that the majority of accidents occurred where the local conditions could be considered favourable for the safety of the pedestrian. As can be seen 85% of accidents occurred on a sealed dry road, 52% in daylight, and 81% in clear weather conditions. Overall 12 of the 27 (44%) accidents took place on a sealed dry road, in clear sunshine, conditions that would be considered as perfect for the safety of

the pedestrian.

Fig 3.1 also shows 4 of the 27 (approx 15%) accidents occurred in the rain. A further study of these accidents revealed that 3 of these occurred when the lighting conditions were classified as lit.

It would be expected that the prevailing weather conditions would dictate the number of pedestrians on the road. In adverse conditions such as rainy weather, it would be highly likely that there would be very few people travelling on foot, and certainly a great reduction from the number to be walking in clear weather.

Coupled with the fact that on average it is raining for less than 10% of the year, it tends to indicate the risk of a pedestrian being involved in an accident in wet weather is much greater than the risk in clear dry conditions.

Given that 75% of these accidents that occurred in wet weather occurred under lighted conditions, it is reasonable to assume that there is a very high risk to the pedestrian in rainy weather at night.

LOCAL GEOMETRY	NUMBER OF ACCIDENTS	% OF TOTAL ACCIDENTS
HORIZONTAL FEATURES		
Straight	22	81.5
Curve - view obscured	0	0.0
Curve - view open	5	18.5
VERTICAL FEATURES		
Level	18	66.7
Slight Grade	9	33.3
Steep Grade	0	0.0
Crest	0	0.0
Dip	0	0.0

FIG 3.2 - Local Geometry at each Accident Site

Fig 3.2 details the horizontal and vertical features of the accident location. This information was collected after an inspection of each of the different sites where the accidents occurred.

Once again the majority of the accidents occurred where the local geometry of the road could not be considered contributary to the accident. 81% of the accidents occurred on a straight section and the remainder on a curve with an unobstructed view for the driver.

Two-thirds occurred on a level road and the other third on a road with a slight grade.

Overall 16 of the 27 accidents occurred on straight, level sections of roads, and when combining the statistics with the local conditions a quarter of the total accidents occurred on a straight, flat, sealed dry road in clear sunshine, the ideal environment for the safety of the pedestrian.

This is the highest percentage for any combination of local factors, and generally this combination should be the safest for the pedestrian.

3.2 Time of Accident

This section looks at the hour, day, month, and seasonal occurrence of the accidents. Figs 3.3 to 3.6 summarise the data.

Friday is the most likely day of the week accidents will occur on. It is a fairly random distribution of days and it does not lead to producing anything conclusive about the relationship of days of the week to accidents. It is interesting to know that every accident on the weekend occurred between 6 PM and 3 AM. This indicates the risk is higher at nights on weekends.

Fig 3.6 peaks at 3 PM and 8 PM. The former time corresponds

DAYS OF WEEK	NUMBER
Monday	5
Tuesday	2
Wednesday	4
Thursday	2
Friday	6
Saturday	4
Sunday	4

FIG 3.3

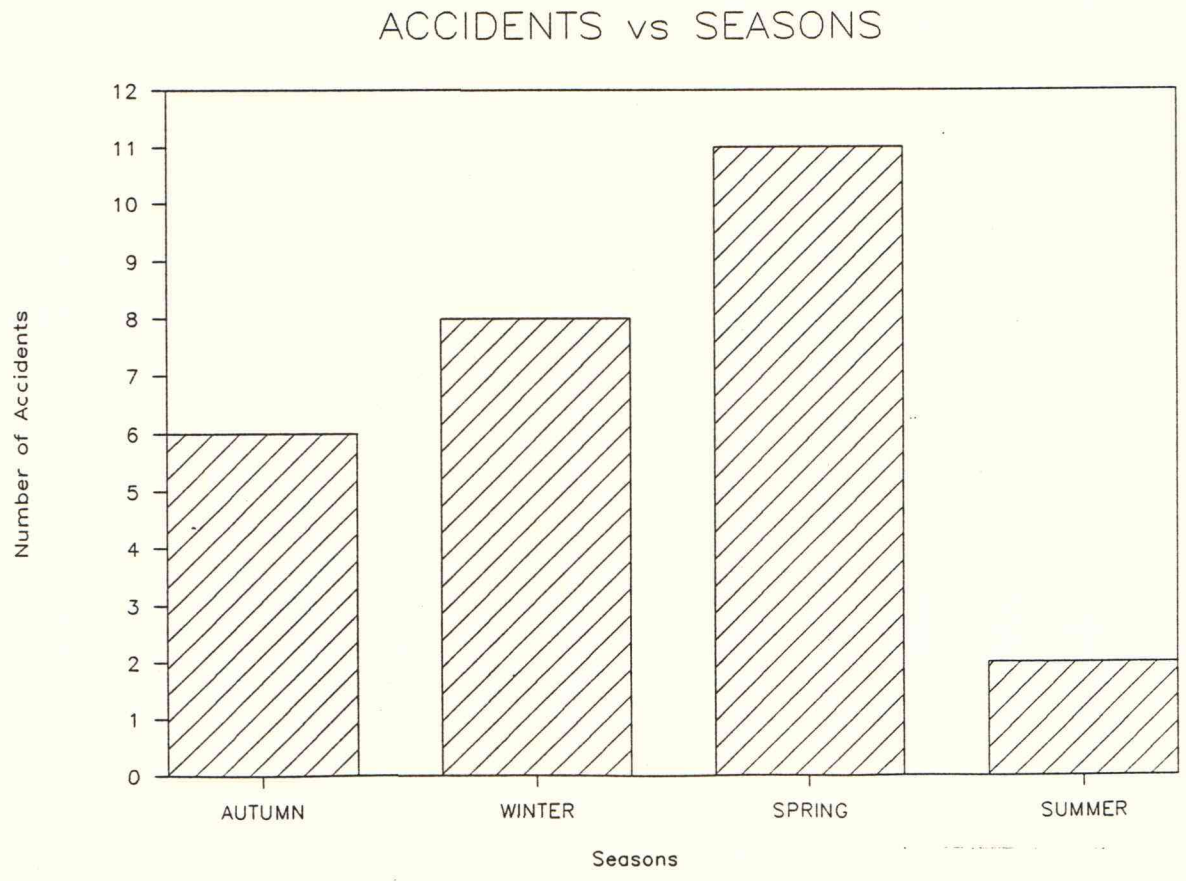


FIG 3.4

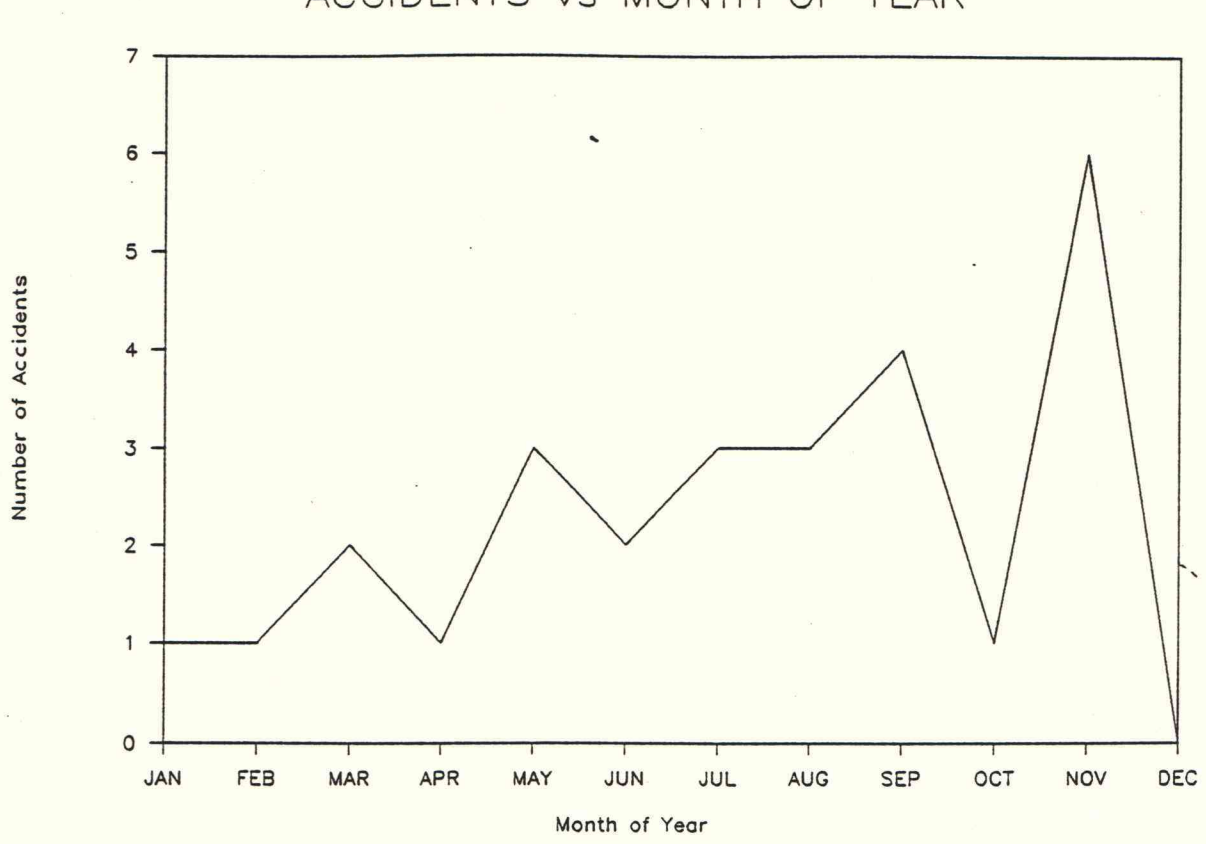

ACCIDENTS vs MONTH OF YEAR
Number of Accidents
0
1
2
3
4
5
6
7
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
Month of Year

FIG 3.5

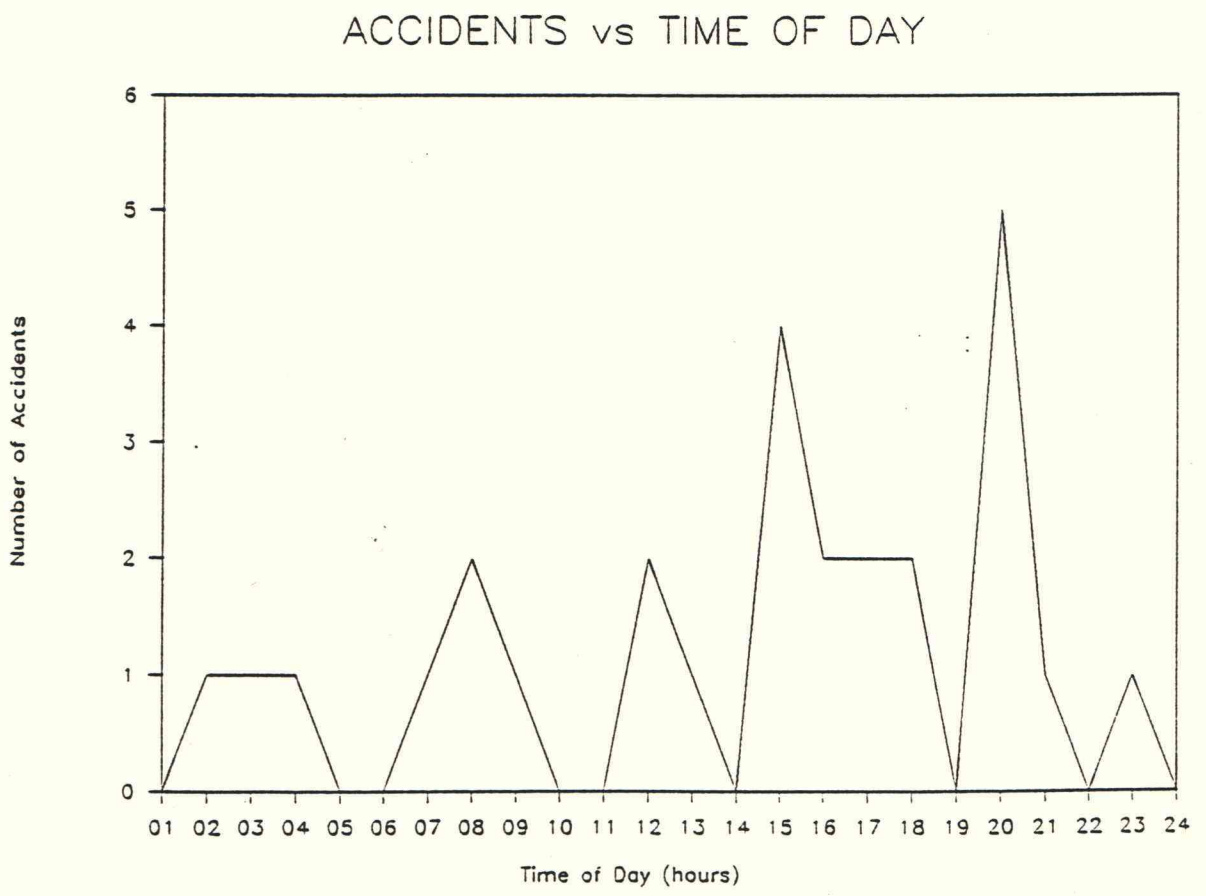

ACCIDENTS vs TIME OF DAY
Number of Accidents
0
1
2
3
4
5
6
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Time of Day (hours)

FIG 3.6

to the time students finish school and when the pedestrian volume would increase.

The majority of accidents occur in spring which would be the most favourable time of the year for pedestrian travel. An increase in numbers would occur during spring and a decrease during summer, when the accident rate is lowest.

Overall very little else can be drawn from this information.

3.3 Location of Accidents

The accident location is described by four factors; traffic control, special features, type of road, and location, and these are summarised in Fig 3.7.

ACCIDENT LOCATION	NUMBER OF ACCIDENTS	% OF TOTAL ACCIDENTS
TRAFFIC CONTROL		
Traffic Lights	4	14.8
Stop Sign	7	25.9
Give Way Sign	4	14.8
No Sign or Control	12	44.4
SPECIAL FEATURES		
Cross intersection	12	44.4
T intersection	2	7.4
Y intersection	1	3.7
Bridge	3	11.1
No special features	9	33.3
TYPE OF ROAD		
Divided	24	88.9
Undivided	3	11.1
LOCATION		
On carriageway	26	96.3
Off carriageway	1	3.7

Fig 3.7 - Location of each Accident

When examining the type of road the accidents took place upon, it must be remembered that the data gathered was for accidents occurring on state declared (Main Roads) roads in Rockhampton, the majority being divided, with Lakes Creek Road, the Fitzroy Bridge, and Queen Elizabeth Drive the only exceptions. It would therefore be an inaccurate assessment of this statistic to take it as representative of Rockhampton.

The declared roads aside, there are very few other divided streets in the city. In the Central Business District, William and Denham Streets are divided and these carry a high pedestrian load. Other divided streets are: Moores Creek Rd, (a section of)

High St, Richardson Rd (east of Yaamba Rd), Feez St, Penlington St, (a section of) North St. All however are not considered major pedestrianized areas.

Considering 33% of the total accidents occurred on the declared roads, it would be highly likely that the majority of accidents occurred on undivided sections of road in the city.

The other factors to be considered are the control of traffic at the accident sites, and the prevalence of intersections in the figures. All collisions, except one occurred on the carriageway, indicating the pedestrian's safety is virtually guaranteed when using the footpath.

Approximately 15 (55%) of the accidents took place where a form of traffic control was present. Of these seven were at STOP signs; and four each at GIVE WAY signs and traffic lights. Generally traffic lights are placed at the most hazardous sites. The data clearly shows that the traffic lights do reduce the risk of danger to the pedestrian.

With 11 of the accidents occurring at GIVE WAY or STOP signs, it is likely that confusion over right of way between driver and pedestrian is the major reason behind the collisions.

Fig 3.7 also shows a higher percentage of accidents occur at intersections as opposed to anywhere else. This is in line with the Australian figure of 65% (although slightly lower - 55%).

Cross intersections account for 12 of the 15 accidents, with two at T-junctions, and one at a Y-intersection.

Three accidents occurred on the two bridges in Rockhampton across the Fitzroy River and nine occurred where no special features were present. It is interesting to know that three of the four deaths occurred where no special features existed and the fourth on the Neville Hewitt Bridge.

So although the chances of an accident occurring are higher at intersections, you are more likely to be killed away from them when struck by a vehicle.

3.3.1 Blackspots

Although the data is only for declared roads in the city, a few locations had more than one accident occurring at them, or in close proximity to another. Fig 3.8 is a map highlighting where the accidents took place in Rockhampton.

Three accidents took place at the intersection of Fitzroy and Kent Streets, two on the northern end of the Neville Hewitt Bridge above Glenmore Road, two on George Street - 180m north of the Fitzroy Street intersection, and two on Gladstone Road near the "Wagon Wheel" service station and restaurant.

Interestingly nine accidents occurred in a 1.5km section along Fitzroy St, from George St to a point on the bridge. Every intersection from George Street to East Street, excepting Alma Street, had at least one accident occur.

The two accidents occurring in George Street were under lit conditions, suggesting the lighting may be inadequate for the safety of the pedestrian in this location.

The pedestrian was at fault on both occasions in the collisions on the bridge, but on this section, there is no seperation of pedestrian and vehicular traffic, as there is for the majority of the bridge.

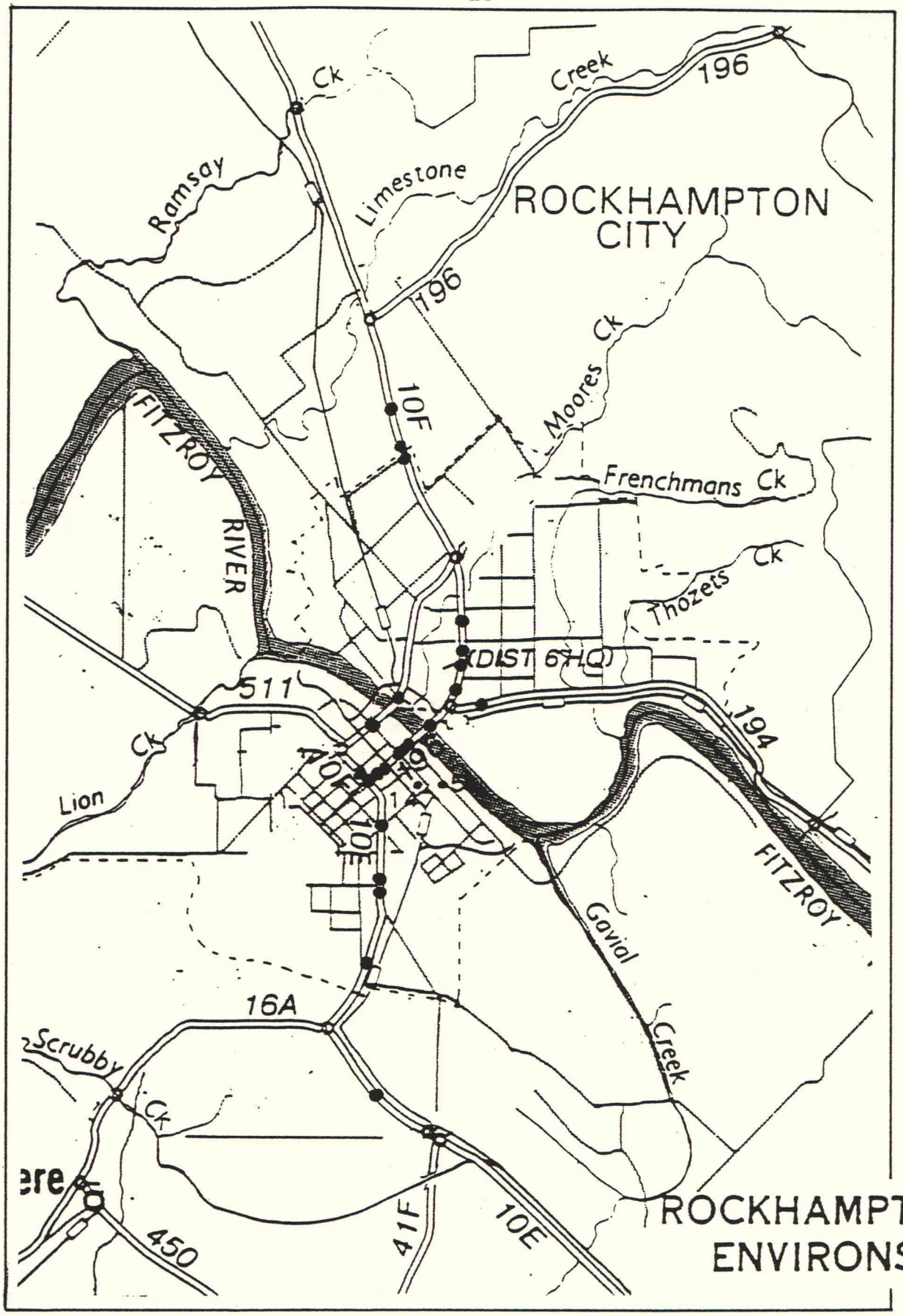
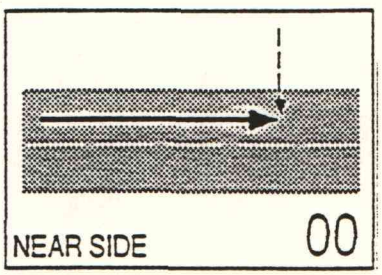


FIG 3.8 - Location of Accidents in Rockhampton

● denotes accidents

3.4 Nature of Accident

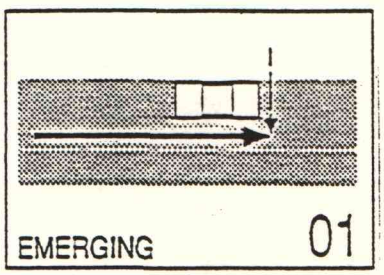
The nature of the accident is described by the use of a ROAD USER MOVEMENT or (RUM) Number, which is a two digit coded figure summarising the nature of impact when the accident occurred. The Department of Transport use these input codes with the PHYLAK accident data base program. For accidents involving pedestrians numbers 00 to 07 are used and these codes are explained below:



00..NEAR SIDE

Pedestrian proceeds from kerb or side of carriageway and is hit by a vehicle from the right - ie. travelling on the side closer to the pedestrian.

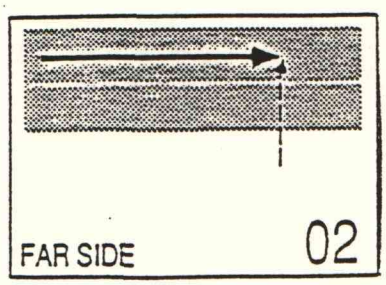
- pedestrian may be in the process of crossing but actually stationary at the time
- vehicle may be reversing
- use this code if the direction of travel of the pedestrian is unknown



01..EMERGING

As above, but pedestrian comes from in front of a parked or stationary vehicle (not a bicycle) on the carriageway - ie on left side of road.

- pedestrian may be emerging from either side of the road on a one way street
- if at an intersection, vehicle may be turning right or left



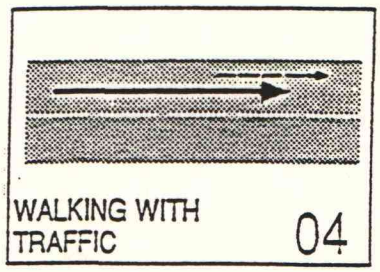
02..FAR SIDE

Pedestrian proceeds from kerb or side of carriageway to cross the road and is hit by a vehicle from the left - ie. travelling on the side of the road furthest from the pedestrian.

- may be emerging from the vicinity of parked or stationary vehicle

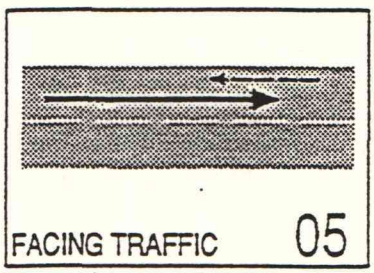
03..PLAYING, WORKING, LYING, STANDING ON CARRIAGEWAY

Pedestrian playing, working, lying, standing etc on the carriageway. It does not include any pedestrian who was in the process of crossing the road at the time.



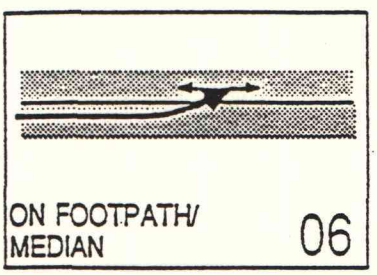
04..WALKING WITH TRAFFIC

Pedestrian is walking on carriageway with the line of traffic when struck by a vehicle.



05..FACING TRAFFIC

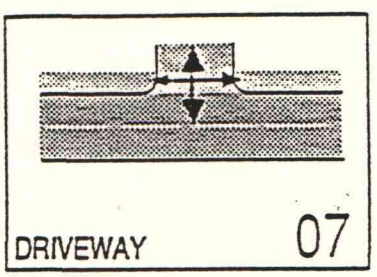
Pedestrian is walking on carriageway against the line of traffic (ie. facing) when struck by a vehicle.



06..ON FOOTPATH/MEDIAN

Vehicle strikes pedestrian on footpath, median or traffic island etc:

- vehicle may run off the carriageway or be travelling along the footpath but NOT leaving or entering a driveway



07..DRIVEWAY

Vehicle entering or leaving driveway strikes pedestrian on footpath.

Fig 3.9 shows the number of accidents that are credited to each respective RUM number. The statistic that stands out from this data is that 78% of the accidents were classified as 00, 01 02 RUM number accidents. Of this, blame could only be attributed to drivers on 4 of the occasions, the rest being accountable to

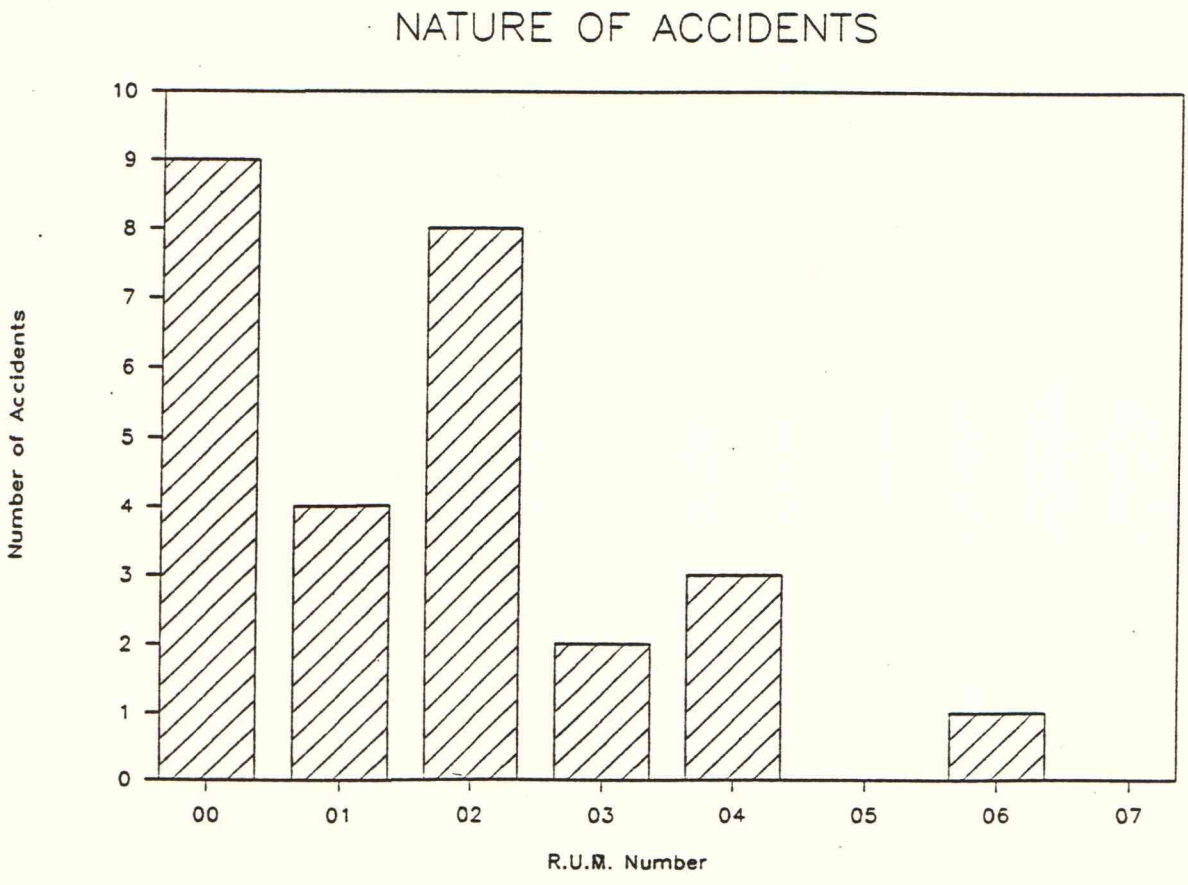


FIG 3.9 - Nature of Accidents by R.U.M. Numbers

the pedestrian. These type of accidents are the result of a lack of due care and attention. In this case a little more care could easily have prevented the accidents. Of the three accidents that occurred under category 04, one occurred on the Fitzroy River Bridge where a footpath is provided, one occurred 60m north of the Farm St - Yaamba Rd intersection where there is very little shoulder room on the road formation due to relatively steep embankment, and the third occurred at the Fitzroy St - Campbell St intersection where adequate footpath widths are provided. Two of the accidents could have been avoided by the victim using the footpaths provided, but the third was a result of inadequate facilities for pedestrians on the stated road. The pedestrian had little choice but to use the carriageway.

3.5 Speed Limit

Speed zones do not seem to have an effect on the number of accidents occurring in the city. Only one of the accidents occurred in a 100 km/h zone, three in 80 km/h zones, and the remainder in 60 km/h zones. These figures are quite understandable as the majority of the roads are 60 km/h zones in the city, with few 80 km/h regions, and even less 100 km/h zones. So these statistics tend to correlate well with one another and do not present the speed limit as an influencing factor in the accidents occurring.

3.6 Driver Licences

The data showed that 7 of the 27 (26%) accidents involved drivers of vehicles that were not licenced to drive at the time. This seemed to be a fairly high proportion, and after consultation with the Police, Department of Transport, and the

Government Statisticians Office, it was estimated that 5-10% of drivers on the road were not licenced to drive. This figure shows that you are much more likely to be involved in a pedestrian accident if you are not licenced to drive.

Ironically, in all of the 7 accidents, fault was attributed to the driver of the vehicle.

3.7 Vehicles Involved

Fig 3.10 shows the breakdown of the type of vehicles involved and the blame attributed to each category.

VEHICLES	INVOLVED	AT FAULT
	NUMBER OF ACCIDENTS	NUMBER OF ACCIDENTS
Car , Station Wagon	17	4
Utility , Panel Van	3	1
Motorcycle	3	1
Truck	2	1
Articulated Vehicle	1	1
Omnibus	1	0
TOTAL	27	27

FIG 3.10 - Breakdown of the Vehicles Involved in Accidents

Passenger vehicles (cars, station wagons) are by far the most common of the vehicle classes involved in accidents. Just on a quarter of these vehicles were at fault.

Motorcycles, Utilities, and Panel Vans were involved in 10% of the accidents, and are the most prevalent after passenger vehicles. This is of some concern as motorcyclists are also highly exposed to injury when striking a pedestrian. Only one third of these motorcyclists were at fault.

All other vehicles are represented but in minimal numbers, and nothing conclusive can be drawn from the remainder of the data.

3.8 Injuries

Fig 3.11 shows the breakdown of the extent of injuries received by the pedestrian in each of the accidents. Unfortunately the actual details and type of injuries sustained were not available from the Department of Transport data.

EXTENT OF INJURIES	NUMBER OF ACCIDENTS	% OF TOTAL ACCIDENTS
Dead	4	14.8
Admitted to Hospital	13	48.1
Treatment at Scene	10	37.0
TOTAL	27	100.0

FIG 3.11 - Extent of Injuries received by Pedestrians

As can be seen 4 of the 27 accidents resulted in death to the pedestrian. The total number of deaths for Rockhampton in the five year period was five, which means 80% of the pedestrian deaths in Rockhampton took place on declared roads. Next to this statistic is the fact that only 33% of the total accidents occurred on declared, indicating that when a pedestrian is struck in the city, he is much more likely to be killed if hit on a declared road as opposed to the rest of the street network.

This could be relevent to the fact the declared roads in general in Rockhampton are the streets where excessive speeding is most likely to occur.

Approximately two-thirds of the pedestrians were at least admitted to hospital, and the remainder were treated at the accident site for some form of injury.

Pedestrians were considered at fault for the four "death" cases and for 11 out of the 13 "hospitalisation" cases.

Conversely, pedestrians in general are not as seriously injured when the driver is to blame for the collision.

4 PEDESTRIAN BEHAVIOUR OBSERVATIONS

4.1 Introduction

A person standing at the side of a busy street is at his own risk to make a judgement on when to cross safely. Generally he makes the right decision but there is that one time in a million when he makes a fatal or near fatal decision. The factors which affect a pedestrian's choice of when it is safe to cross need to be examined in order to reduce that probability of injury or death to the pedestrian.

A study undertaken in the United Kingdom in 1961 revealed the following:

1. Pedestrians make an allowance for the speed of the approaching vehicle, suggesting the pedestrian is more concerned with a time, not distance gap in the traffic.
2. When an approaching vehicle was less than seven seconds away, the pedestrian increased his natural speed of crossing.
3. Economy of the pedestrian's time is the reason for the preference of a time gap over a distance gap.
4. Pedestrians tend to underestimate the following,
   a) speed of vehicles on the far side of the road.
   b) speed of small vehicles (eg. motorcycles).
   c) speed of approaching high speed vehicles.

The study also revealed the relative risk to pedestrians crossing the street for different locations and situations. Fig 4.1 indicates the following with regard to risk factors:

1. The pedestrian is at more risk crossing close to an intersection than he/she is crossing away from them.
2. The pedestrian using a designated crossing is much safer

than he/she crossing away from them.

3. The pedestrian is safer using a signal controlled crossing than a zebra crossing.

LOCATION	TYPE OF CROSSING	RELATIVE RISK
At, or within 20 yards of an intersection	Signal controlled	0.20
	Zebra	0.65
	Elsewhere	1.25
More than 20 yards from an intersection	Zebra	0.22
	Elsewhere	1.00 *

* Arbitrarily taken as unity.

FIG 4.1 - Relative risk to pedestrians.

To determine whether pedestrians are exposing themselves to unnecessary risk, a study of the behaviour of the pedestrian needs to be undertaken in the vicinity of the crossings.

A common measure is the "PEDESTRIAN BEHAVIOUR INDEX". It is:

$$\frac{\text{NUMBER OF PEDESTRIANS USING A CROSSING}}{\text{NUMBER CROSSING ON AND WITHIN 20 YARDS OF THE CROSSING}}$$

Driver behaviour is measured by the "STOPPING INDEX". It is simply the percentage of vehicles giving way to pedestrians, at the crossing. The stopping index is dependent on pedestrian flow and tends to increase with it. As a measure of driver behaviour it is only comparable for similar volumes of pedestrian flow.

Twelve different locations in Rockhampton were studied to determine the general behaviour of pedestrians and to a lesser extent drivers in the city. The scope and results of the observations are presented in the following sections and in Appendix B of this report.

4.2 Locations

The locations were chosen and systematically divided into three categories being:

1. Signal controlled crossings.
2. Zebra crossings.
3. Zebra crossings under "Lollipop Person" control.

SIGNAL CONTROLLED CROSSINGS	UNCONTROLLED ZEBRA XINGS	LOLLIPOP CONTROL ZEBRA XINGS
Fitzroy-East	William-East	Farm St
Fitzroy Bolsover	Fitzroy-Quay	Main St
Bolsover-Denham	Denham-Quay	Upper Dawson Rd
Musgrave-Elphinstone		Berserker St
		Thozet Rd

FIG 4.2 - Locations used for Behaviour Observations

The Central Business District has the highest volume of pedestrian-traffic interaction, therefore six of the sites chosen were located there. The five crossings controlled by "Lollipop Persons" were naturally located adjacent to schools. High traffic volumes, and the tendency for traffic to speed in these areas, were the prime reasons for the choice of these sites. A range of school sizes was chosen to compare the results for various levels of flow.

The study of the Musgrave-Elphinstone intersection was to give behaviour indications in pedestrianized areas away from the Central Business District.

Fig 4.3 and 4.3a show the locations of the "lollipop controlled" crossings, and Musgrave-Elphinstone. Fig 4.4 shows the actual crossing phases and locations studied for the six intersections in the Central Business District.

The "Lollipop Person" controlled crossings were situated away from intersections.

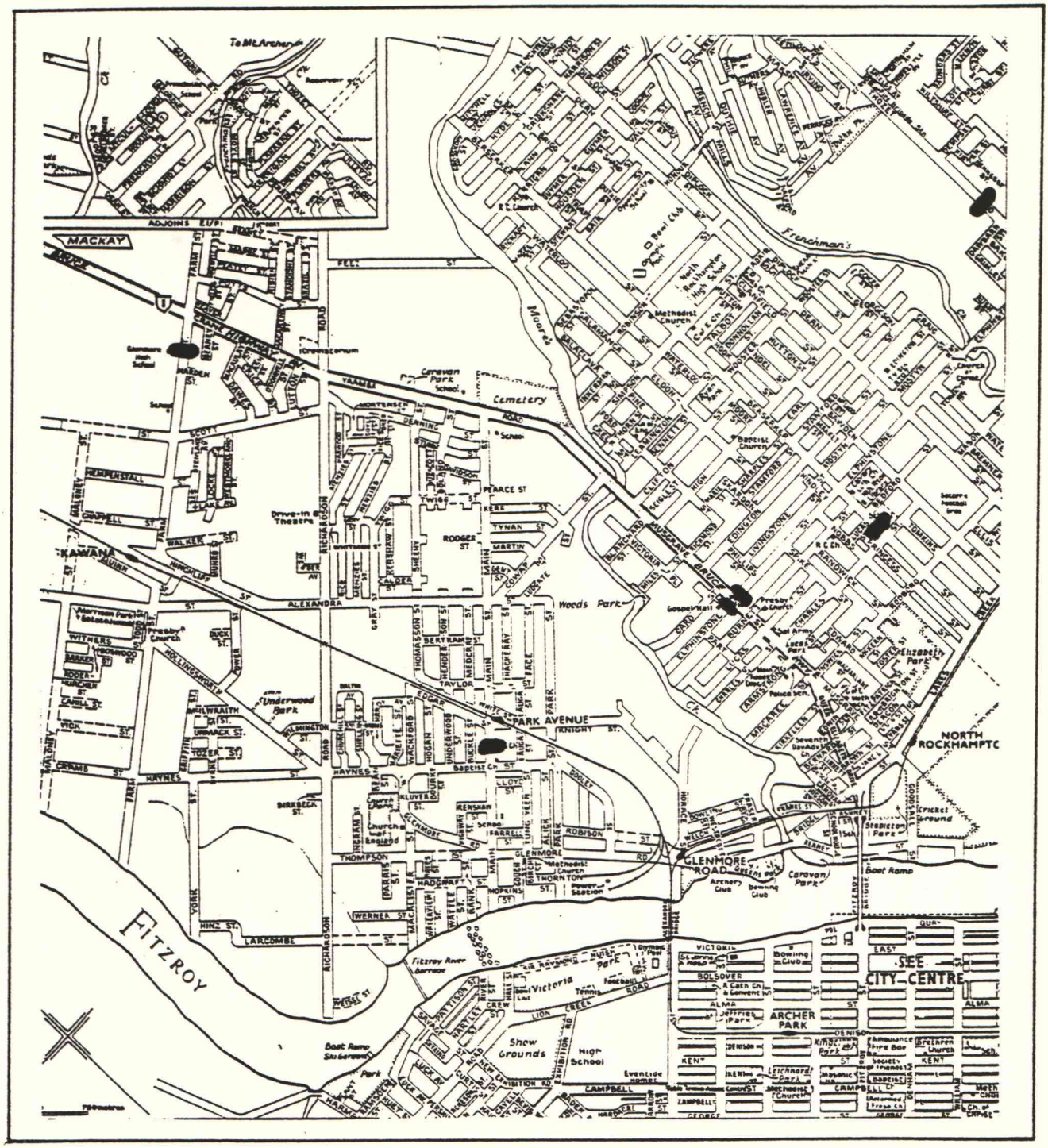


FIG 4.3 - Observation Sites

● denotes crossing

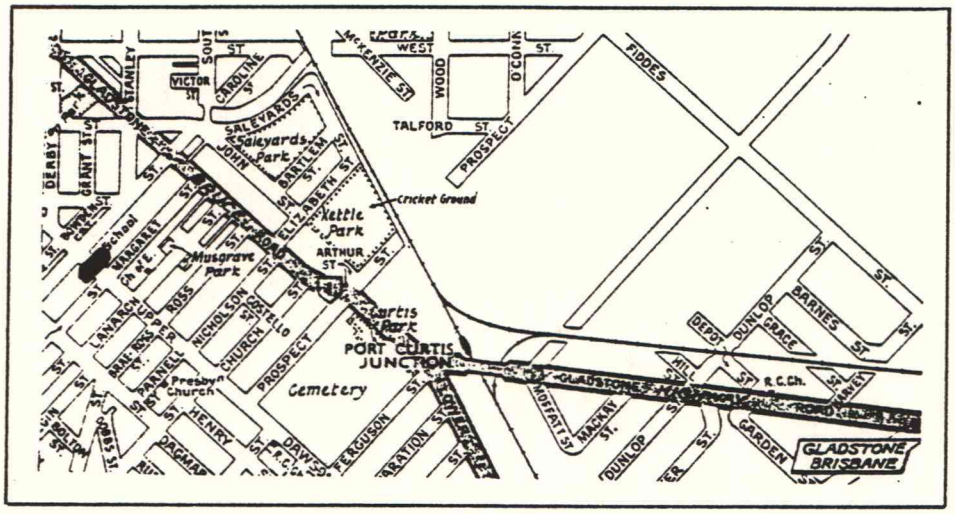


FIG 4.3a - Observation Sites

denotes crossing

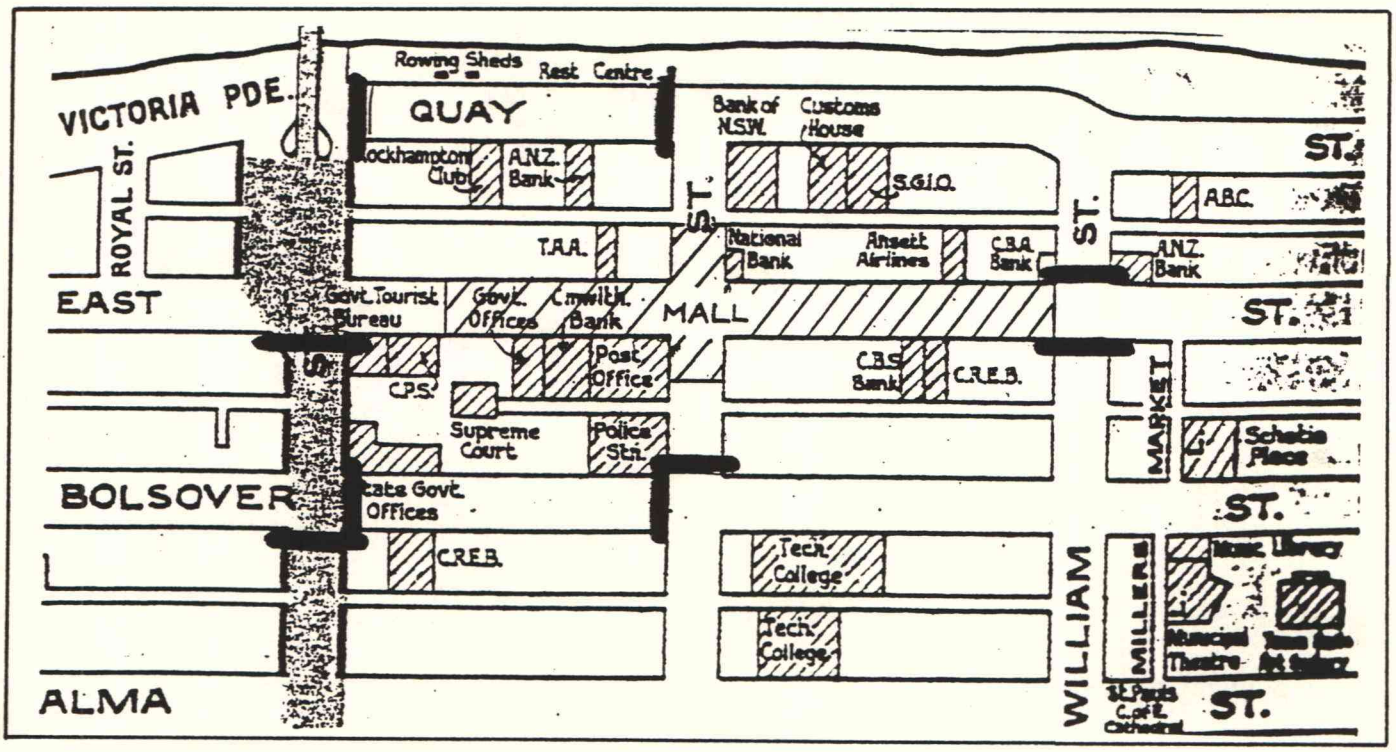


FIG 4.4 - CBD Observation Sites

denotes crossing

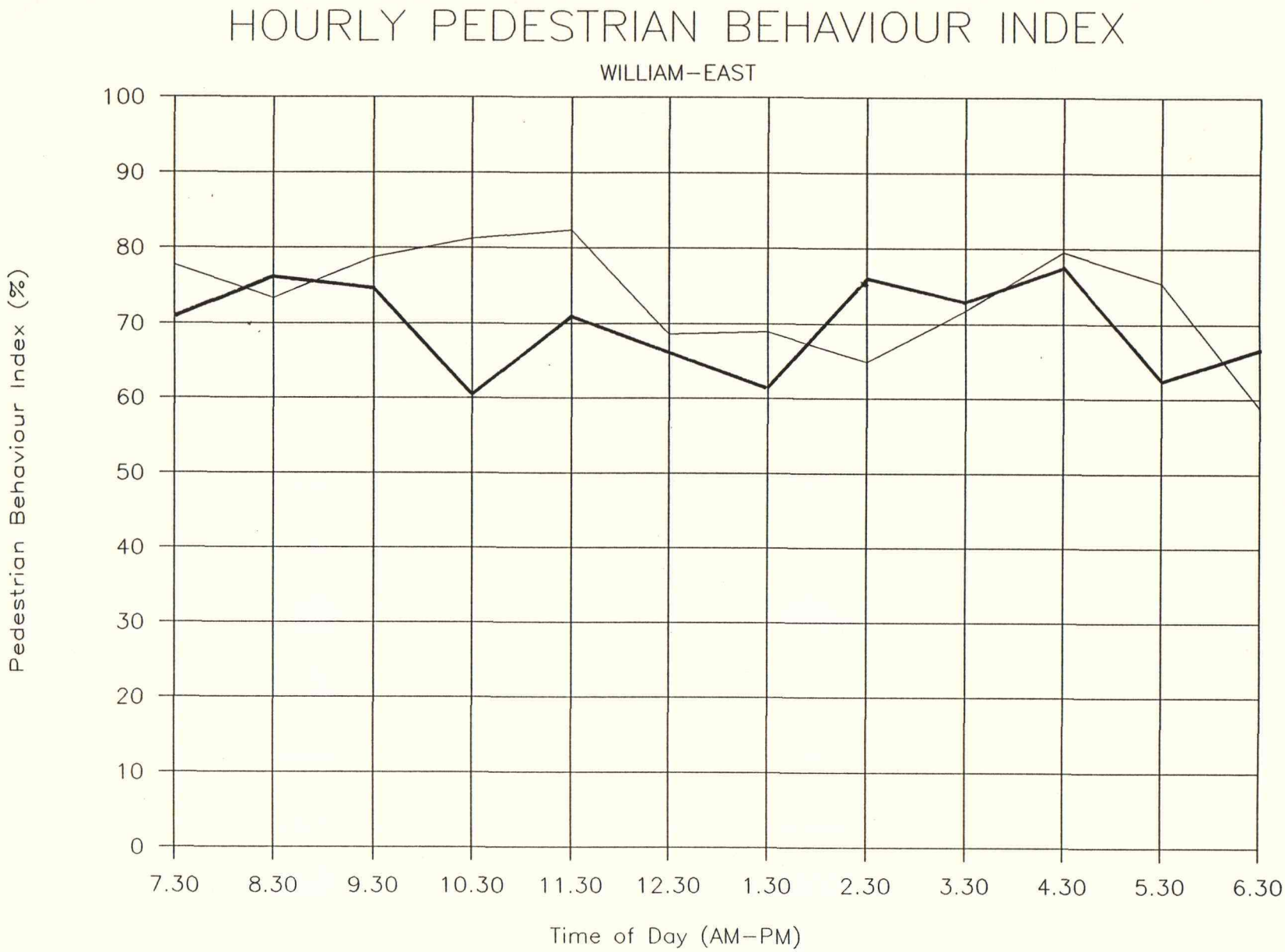

HOURLY PEDESTRIAN BEHAVIOUR INDEX
WILLIAM–EAST
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.5

4.3 Scope

The observations were conducted on Tuesdays, Wednesdays, and Thursdays in the weeks commencing October 1-22 inclusive. These days were considered representative of the pedestrian movement in the city. Studies conducted at the seven intersections were done over a 12 hour period from 7 AM - 7 PM, with results being tabulated at hourly intervals.

The "Lollipop Person" controlled crossings were studied for approximately 2 hours, being when the "Lollipop Person" is on duty. These times were 8 AM - 9 AM and 3 PM - 4 PM approximately.

For all observations data was divided into male and female categories to allow a comparison of the two to be made. General qualitative observations were made during the course of study as well as the quantitative data that was obtained.

The "Stopping Index" was only obtained at the three uncontrolled zebra crossings and was taken over the whole 12 hour period. The number of cars giving way at signallized and controlled zebra crossings is dictated by the form of control that is in operation, therefore no records were kept for these locations.

4.4 Results

The results obtained are presented graphically for the seven intersections in Figs 4.5 to 4.11 (the darker line represents the male curve). The data is summarized in tabulation form in Fig 4.12 and Fig 4.13.

4.5 Discussion of Results

Analysis of the data revealed a number of points about the

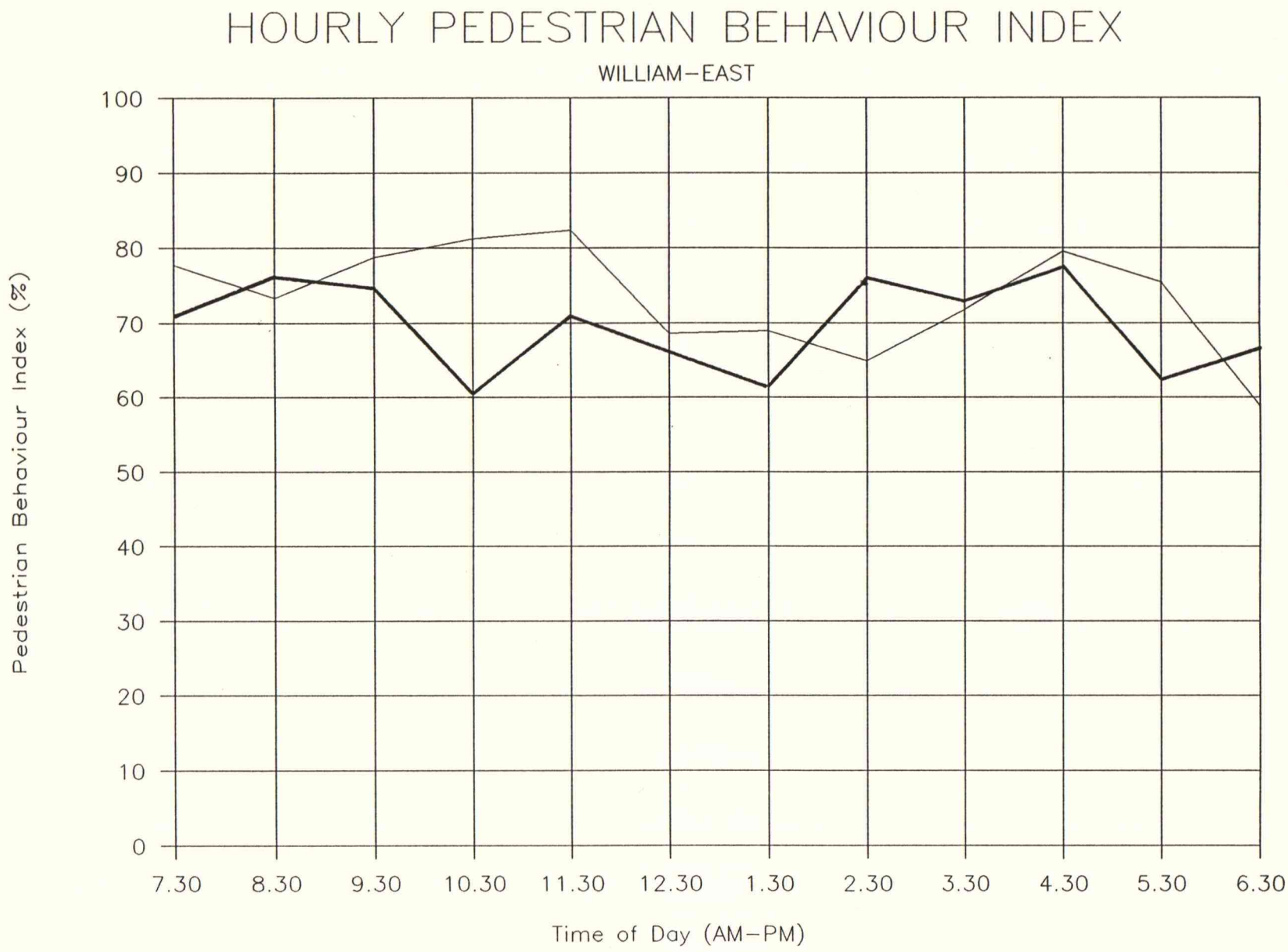

HOURLY PEDESTRIAN BEHAVIOUR INDEX
WILLIAM–EAST
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.5

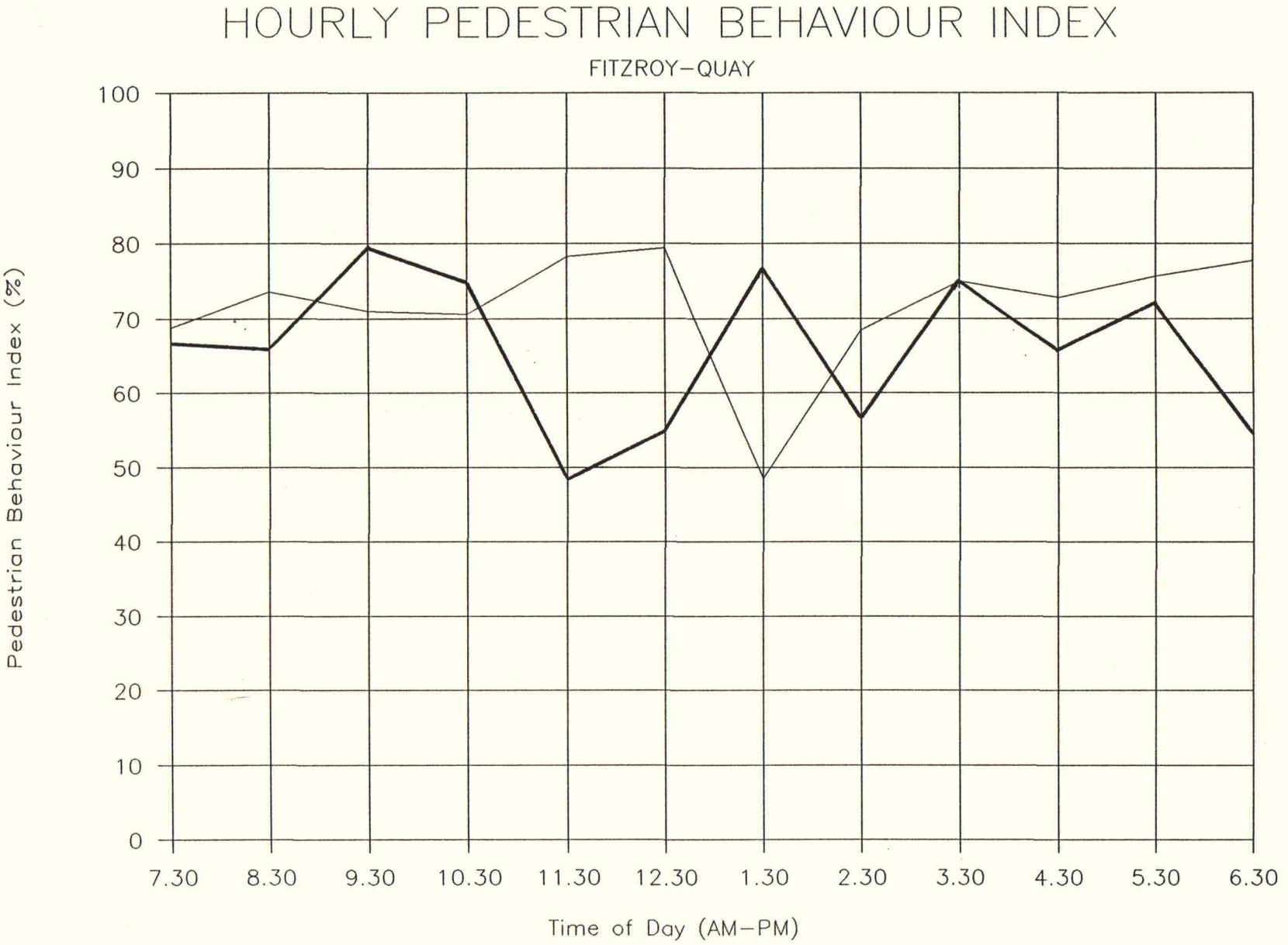

HOURLY PEDESTRIAN BEHAVIOUR INDEX
FITZROY–QUAY
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.6

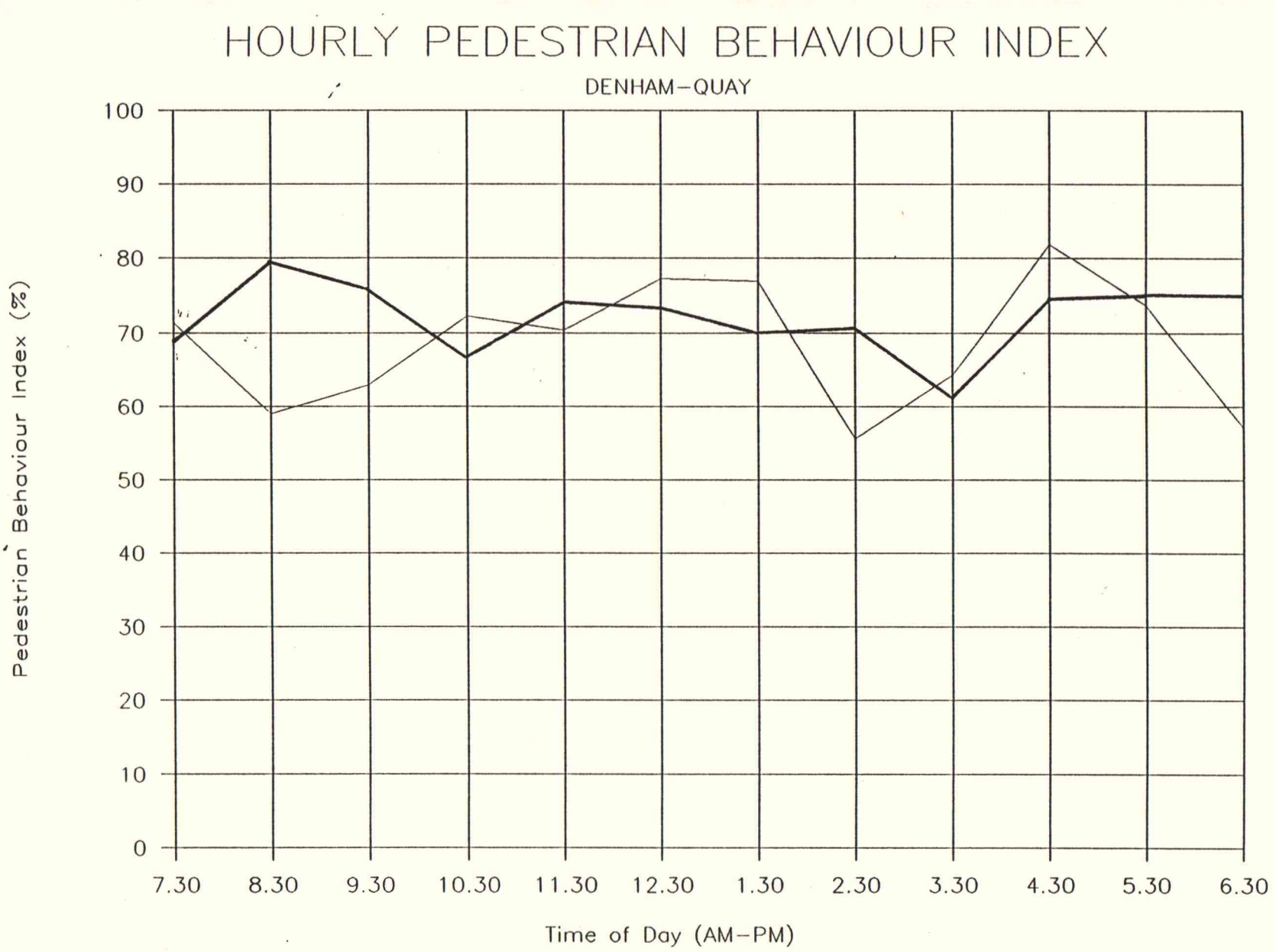

HOURLY PEDESTRIAN BEHAVIOUR INDEX
DENHAM–QUAY
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.7

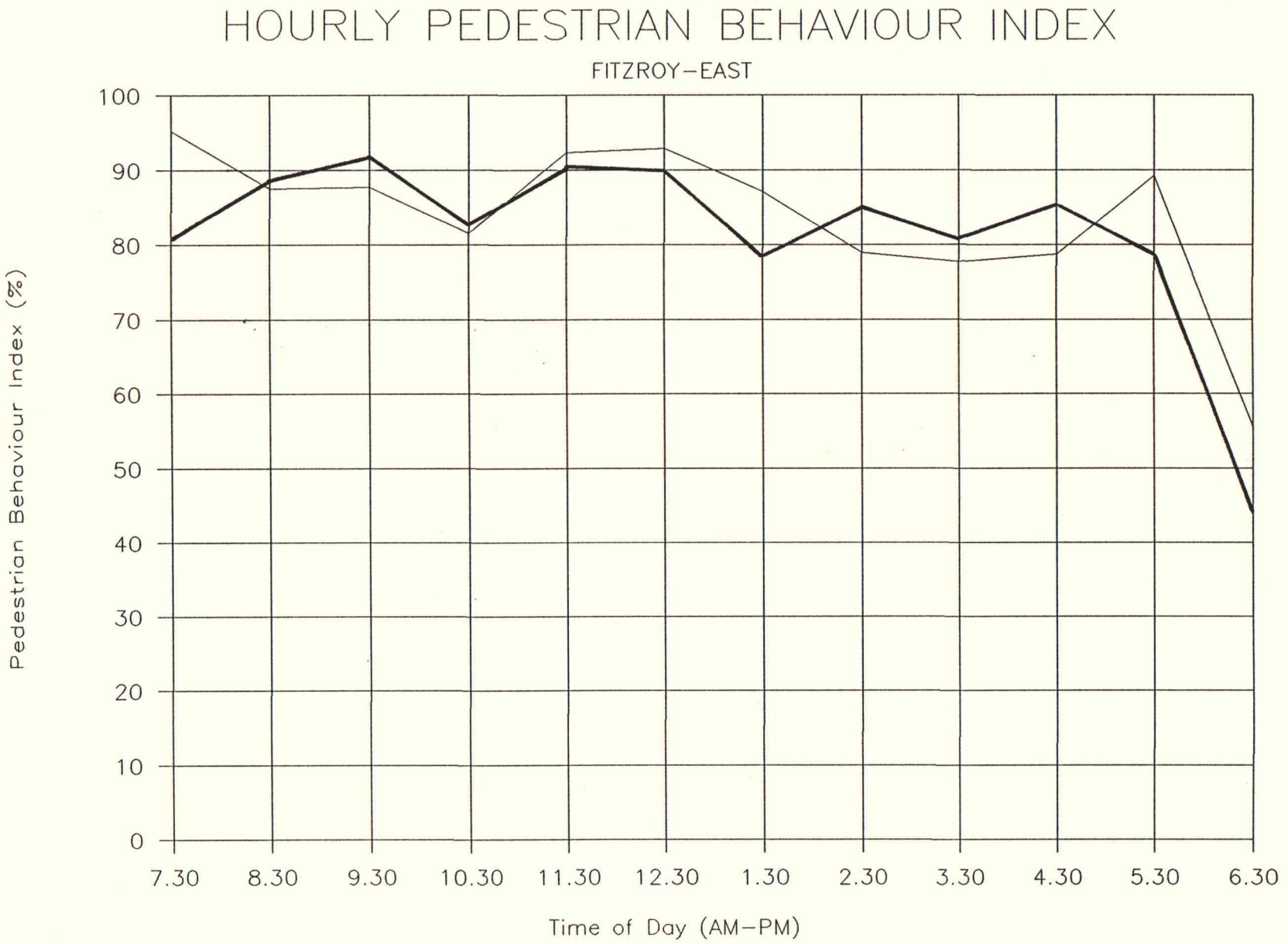

HOURLY PEDESTRIAN BEHAVIOUR INDEX
FITZROY–EAST
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.8

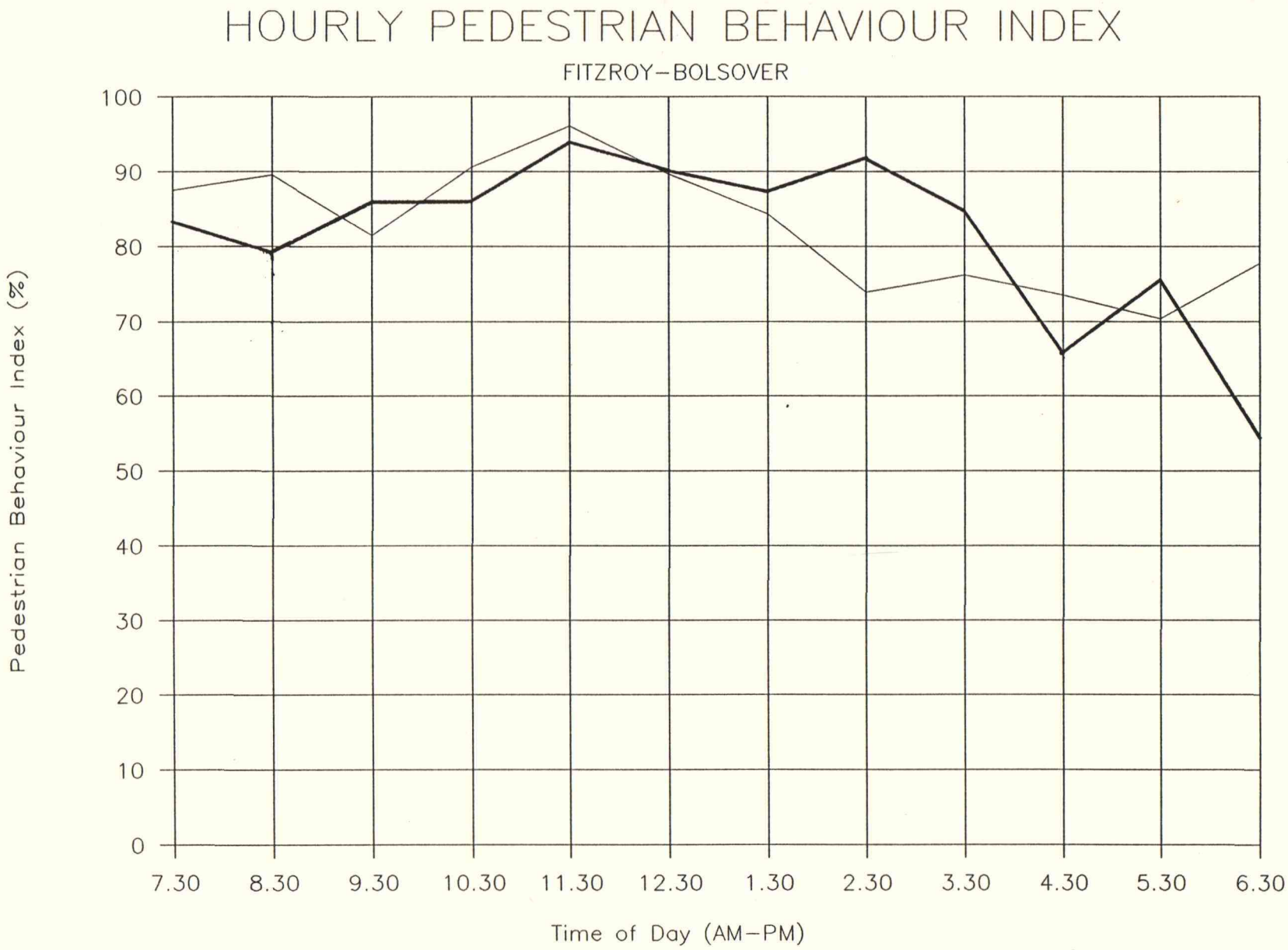

HOURLY PEDESTRIAN BEHAVIOUR INDEX
FITZROY–BOLSOVER
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.9

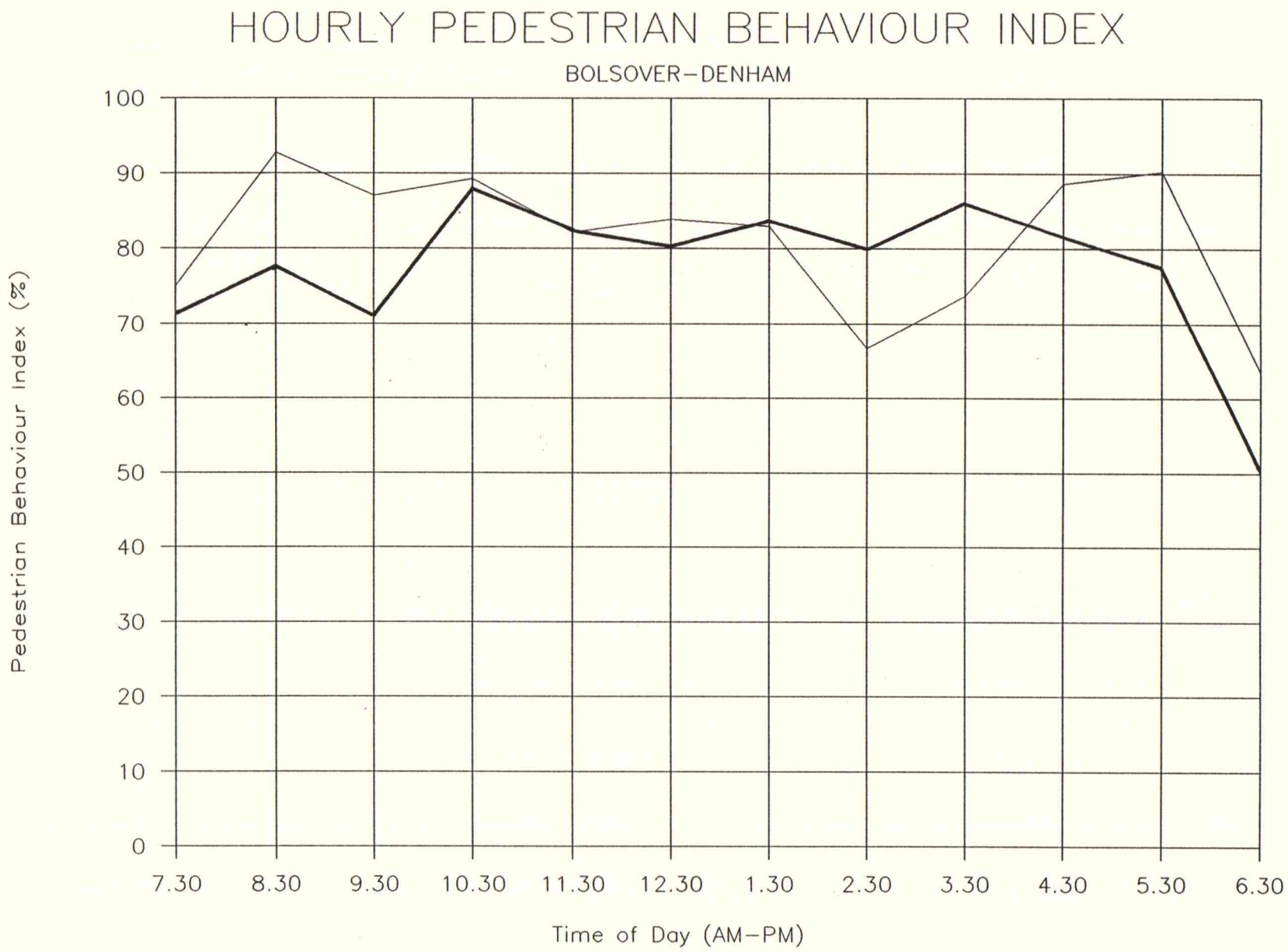

HOURLY PEDESTRIAN BEHAVIOUR INDEX
BOLSOVER–DENHAM
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.10

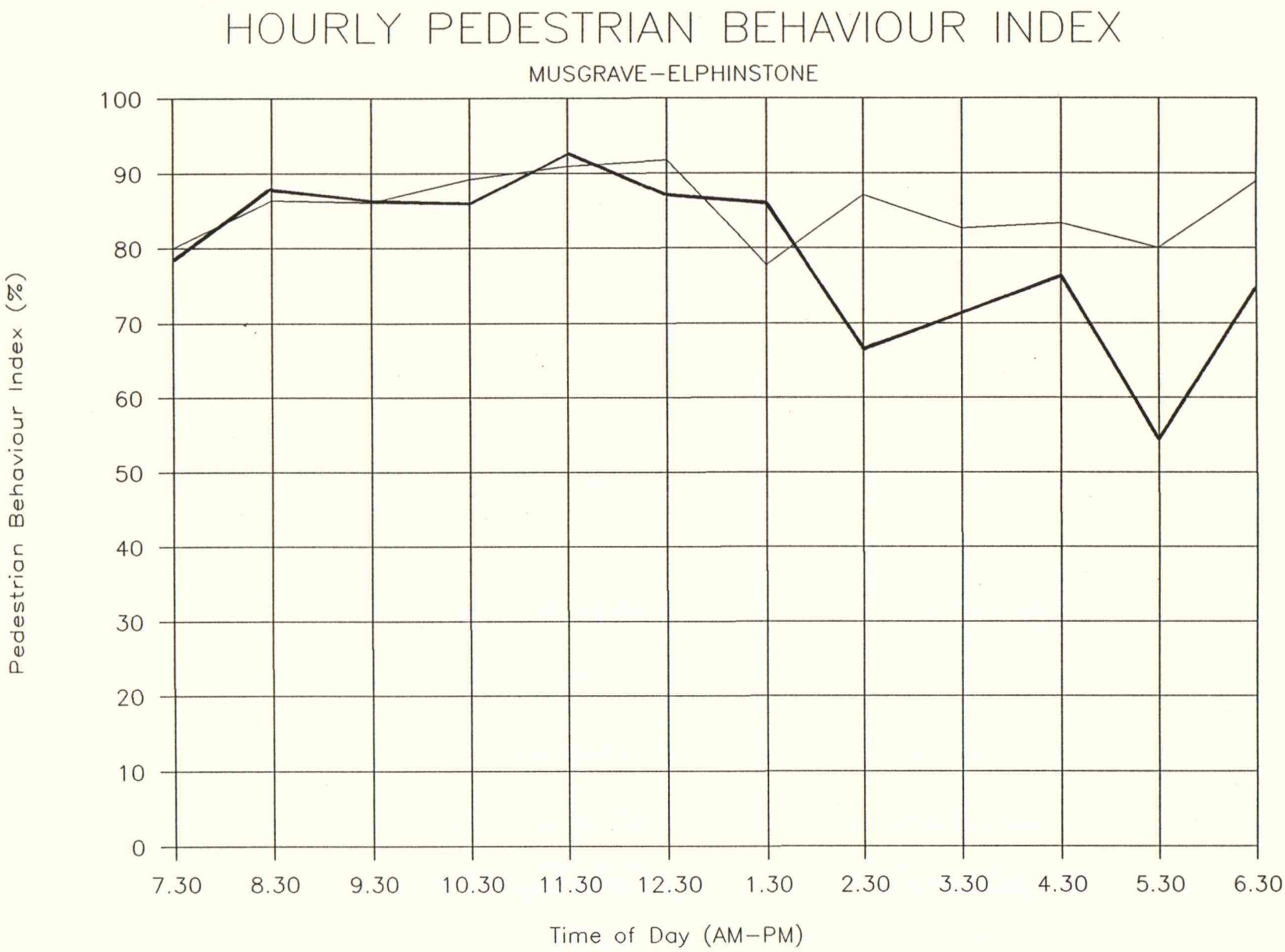

HOURLY PEDESTRIAN BEHAVIOUR INDEX
MUSGRAVE–ELPHINSTONE
Pedestrian Behaviour Index (%)
100
90
80
70
60
50
40
30
20
10
0
7.30
8.30
9.30
10.30
11.30
12.30
1.30
2.30
3.30
4.30
5.30
6.30
Time of Day (AM–PM)

FIG 4.11

INTERSECTION	MALE			FEMALE		
	1	2	BI	1	2	BI
William-East	415	598	69.4	395	532	74.2
Fitzroy-Quay	233	352	66.2	257	360	71.4
Denham-Quay	211	290	72.8	168	242	69.4
Fitzroy-East	375	445	84.3	319	372	85.8
Fitzroy-Bolsover	292	352	83.0	280	333	84.1
Bolsover-Denham	358	453	79.0	351	421	83.4
Musgrave-Elphinstone	261	316	82.6	249	289	86.2

FIG 4.12 - Pedestrian Behaviour Indices for Signal Controlled Crossings and Uncontrolled Zebra Crossings.

CROSSING	MALE			FEMALE		
	1	2	BI	1	2	BI
Farm St	211	237	89.0	298	312	95.5
Main St	81	89	91.0	95	103	92.2
Upper Dawson Rd	131	145	90.3	154	162	95.1
Berserker St	172	188	91.5	176	183	96.2
Thozet Rd	83	93	89.2	109	116	94.0

FIG 4.13 - Pedestrian Behaviour Indices for "Lollipop" Controlled Crossings.

NB In the above figures:

1 = Number of pedestrians using the crossing.

2 = Number of pedestrians crossing within 20 yards of the crossing, and at the crossing itself.

BI = Pedestrian Behaviour Index.

These are daily totals.

behaviour of pedestrians in Rockhampton.

The first and most obvious fact to come out of the survey was that at every site (except Denham-Quay) the behaviour index was higher for women than men. The difference in value ranges from 1.1 through to 6.5 percent but the majority are in the 4 - 5% bracket. This literally means that when people wish to cross the street in the vicinity of a designated crossing, women will generally use the crossings more than men. This could most likely be attributed to a higher level of safety consciousness by women, and an attitude of "doing the right thing".

Secondly the level of pedestrian behaviour is roughly 13% better at signal controlled crossings than uncontrolled zebra crossings, and 10% better in turn at "lollipop controlled" zebra crossings than signal controlled crossings. This is expected as the traffic flow is halted at the controlled intersections. At first it seems a large difference between the lollipop and signal crossings however this could be attributed to the fact that children are under supervision and would be fearful of "getting into trouble" if they did not use the designated crossing. Ideally with children in the high risk bracket a 100% figure should not be out of the question and it was noticed the children who did cross away from the crossing were given a reprimand by the controlling officer on most occasions.

Thirdly Figs 4.5 to 4.11 give away very little about the level of pedestrian behaviour in relation to the time of day. The curves don't seem to follow any real pattern. The behaviour index seems to remain fairly consistent within 20% generally for most of the day at signal controlled crossings before tapering off towards the end. The index tends to be lower during the busy periods (approx 8 AM - 9 AM and 5 PM - 6 PM), in the afternoon.

This is most likely a result of people hurrying to get home taking the shortest way across the street after work, and thus are too lazy to walk to the crossing. Nothing conclusive can be drawn from the curves on uncontrolled zebra crossings, as they are quite erratic.

The Musgrave-Elphinstone intersection had a male index as high as the city intersections and the fenale index was slightly higher. This indicates there is no real change in pedestrian behaviour from the city to the smaller suburban pedestrianized areas.

The stopping index taken for the three uncontrolled zebra crossings were as follows:

1. William-East 13.5%
2. Fitzroy-Quay 8.7%
3. Denham-Quay 10.6%

The average overall was 11.2% for the three crossings. As previously stated the index can only be comparable for similar levels of pedestrian flow. Unfortunately, the flows were not similar (Fig 4.12) and nothing conclusive can really be drawn. The Fitzroy-Quay pedestrian flows were higher than the Denham-Quay crossing but the stopping index was lower. This could be due to a reduction in sight by the skewed approach under the bridge in Quay Street.

5 PEDESTRIAN SURVEYS

5.1 Introduction

To determine peoples attitude to pedestrian safety in the city of Rockhampton, four independent surveys were undertaken. These surveys were:

1. Schools
2. Adults
3. Neville Hewitt Bridge
4. Fitzroy River Bridge

The schools' and adults' surveys aimed to determine the following features:

1. Frequency of walking.
2. Safety of zebra crossings.
3. Difficulty in negotiating roundabouts.
4. People who had been involved in accidents and the number not reported to police.

The two bridge surveys were carried out to determine people's feelings about the safety of the two bridges to pedestrians. These surveys were on a smaller scale.

Copies of the school and adult survey forms are found in Appendix B at the back of the report.

5.2 School Survey

Approximately 700 forms were distributed to students through the schools, and general public area surveys in the City Heart Mall and Rockhampton Shopping Fair. The breakdown was roughly 100 per year level from years 4-12. Originally year threes were to be included in the study, however one principal advised that the children would be unable to answer the questions rationally.

After some consideration they were dropped from the survey.

The results to the answers of the questions are given in Appendix B. A summary of these results is presented graphically in Figs 5.1 - 5.8.

The first question dealt with determining the number of students who walk to school with any regularity. The number of students who walk to school 3 and 4 days per week was negligible (excepting years 4 and 5, with 10.0% and 18.3%), in comparison with the number for 5 days per week. The percentage of students walking to school 5 days per week (Fig 5.1), varies greatly across the year levels, peaking in year 5 (44%), and then dropping off sharply to a minimum in year 7 (9%). The figure then hovers between 10 - 20% for secondary school students. Comparing males and females, it can be seen from Fig 5.2 that the curve follows similar patterns and values in the primary years, but is fairly random in the high school years. On closer inspection of the graph, an inverse relation is present in years 8 - 12 ie. when the male curve peaks, the female curve troughs, and vice versa. The percentage of males walking to school is greatest in year 8 for high schools, and in year 11 for females. However a lot more primary students walk to school in general than their secondary counterparts.

The second question was designed to gauge the difficulty students find with negotiating an ever increasing traffic control device - roundabouts. Fig 5.3 concisely sums up the results on a whole, but the year level breakdown can be found in Appendix B. The results clearly show that on the whole, over half (58%) of the students do at some time or another encounter difficulties at roundabouts. Males find it easier than females on the whole to cross them and this is true for every year.

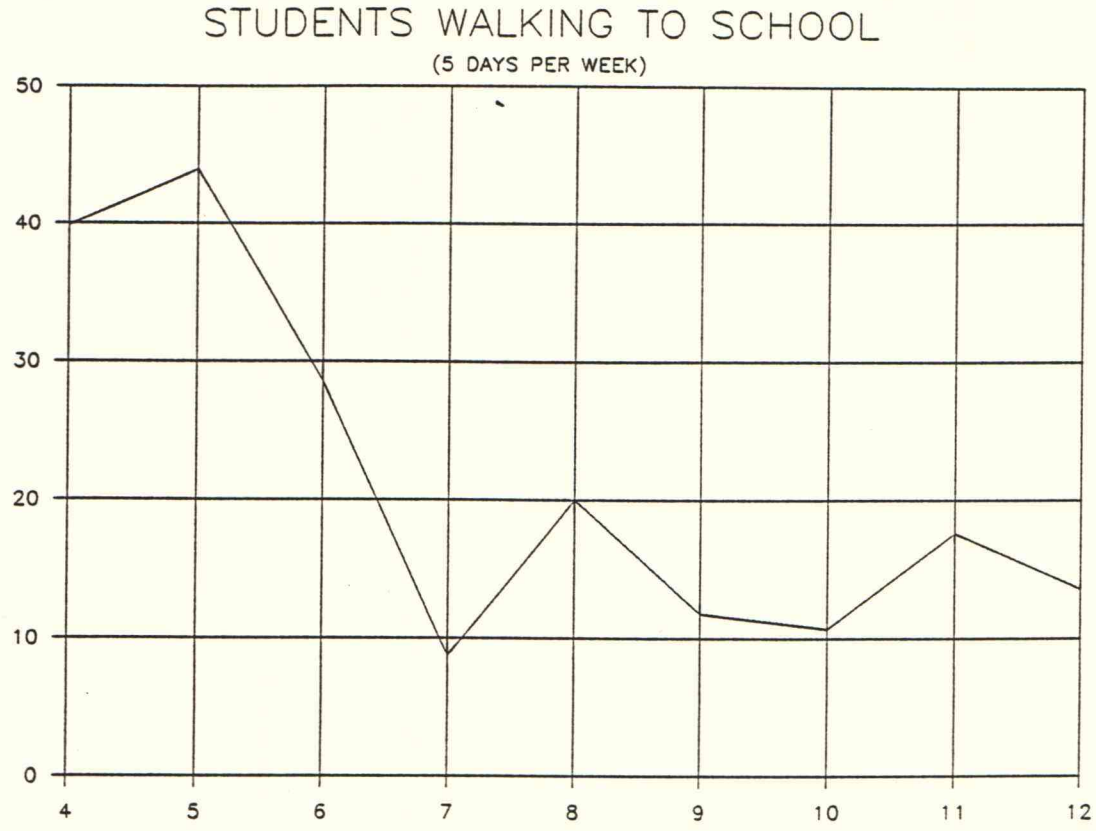


FIG 5.1 - Percentage of Total Students

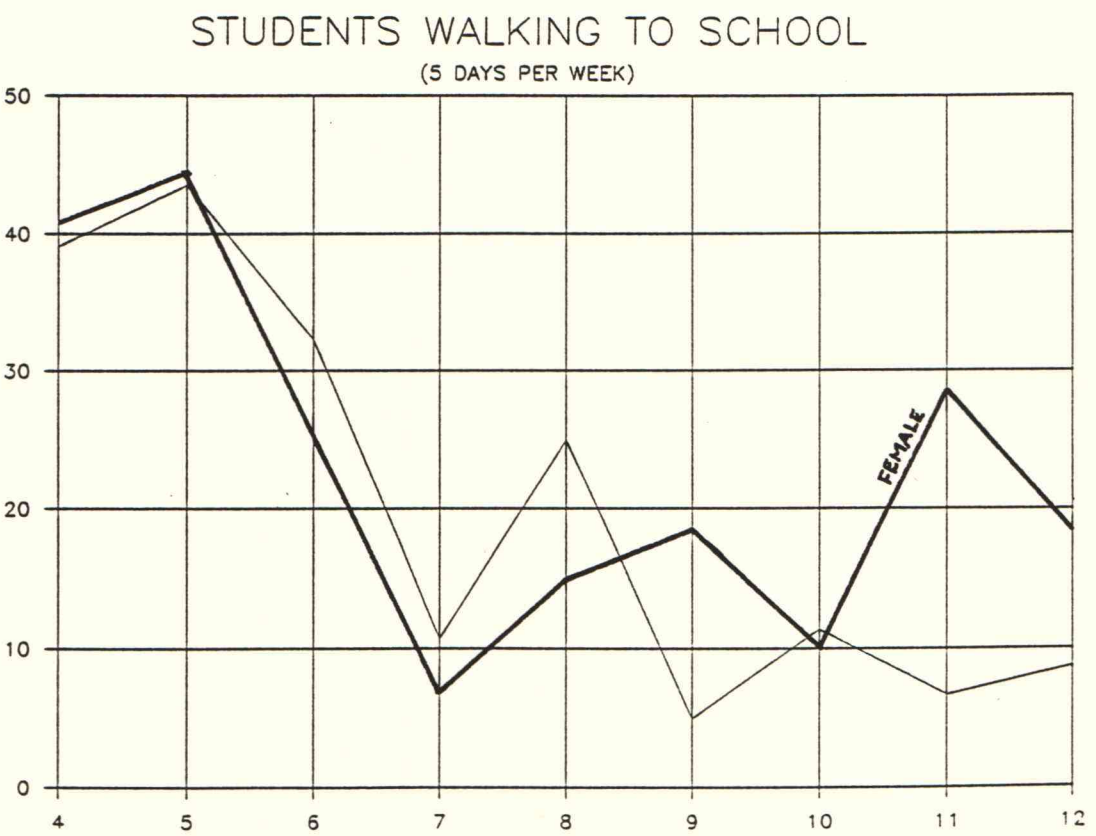


FIG 5.2 - Percentage of Male/Female Students

level. Surprisingly year seven find the least difficulty, while year four and eight find the most. This could be psychologically related with the "big boys and girls" attitude of the children influencing their thinking. Generally speaking, the older male students have less difficulty than the younger ones which is expected, but girls in years 8 and 9 seem to have as much trouble as the ones in years 4 and 5 and certainly a lot more than the remaining years.

Question three is used to give an insight into the students feeling of safety at zebra crossings, both uncontrolled and those controlled by a "lollipop person". Figs 5.4 to 5.7 give the overall summary of the results. As expected, students as a whole regard "lollipop person" controlled intersections much

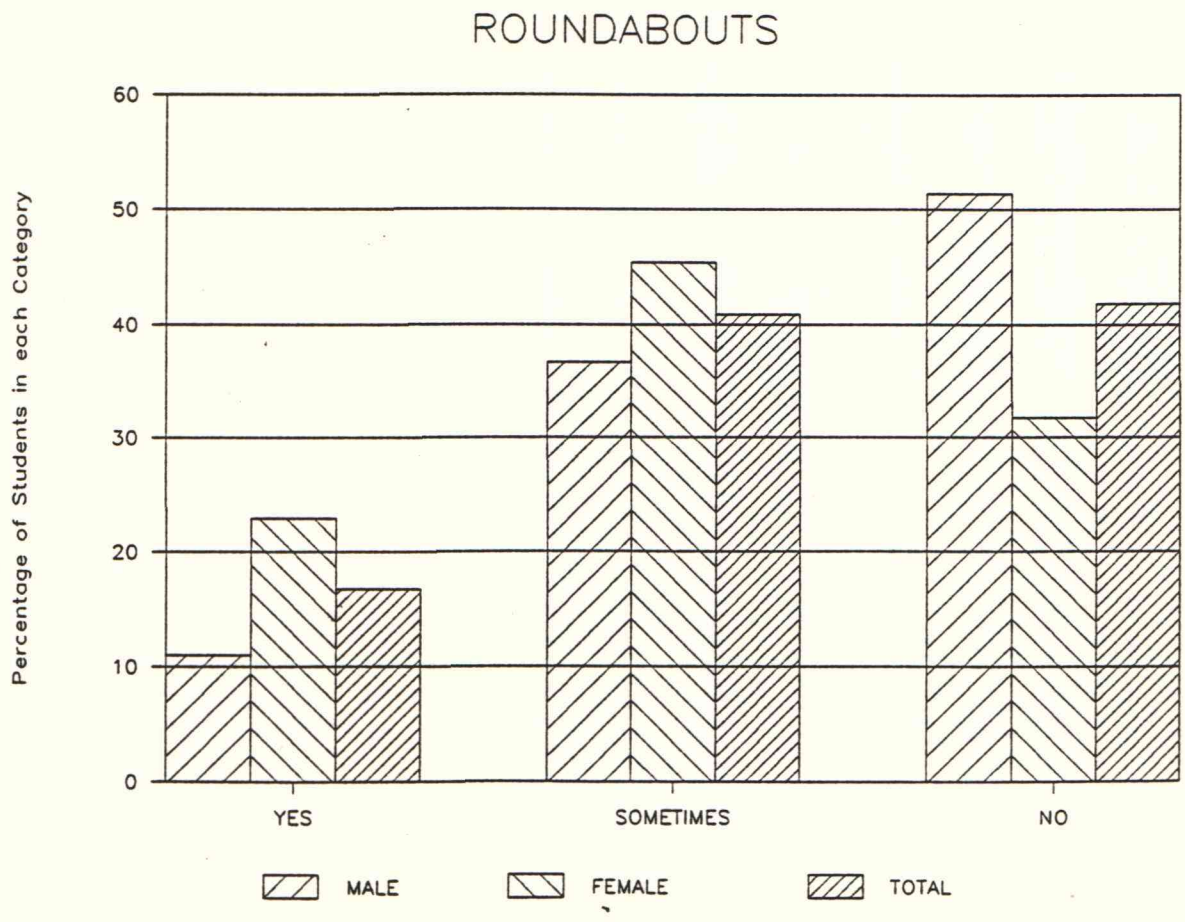


FIG 5.3

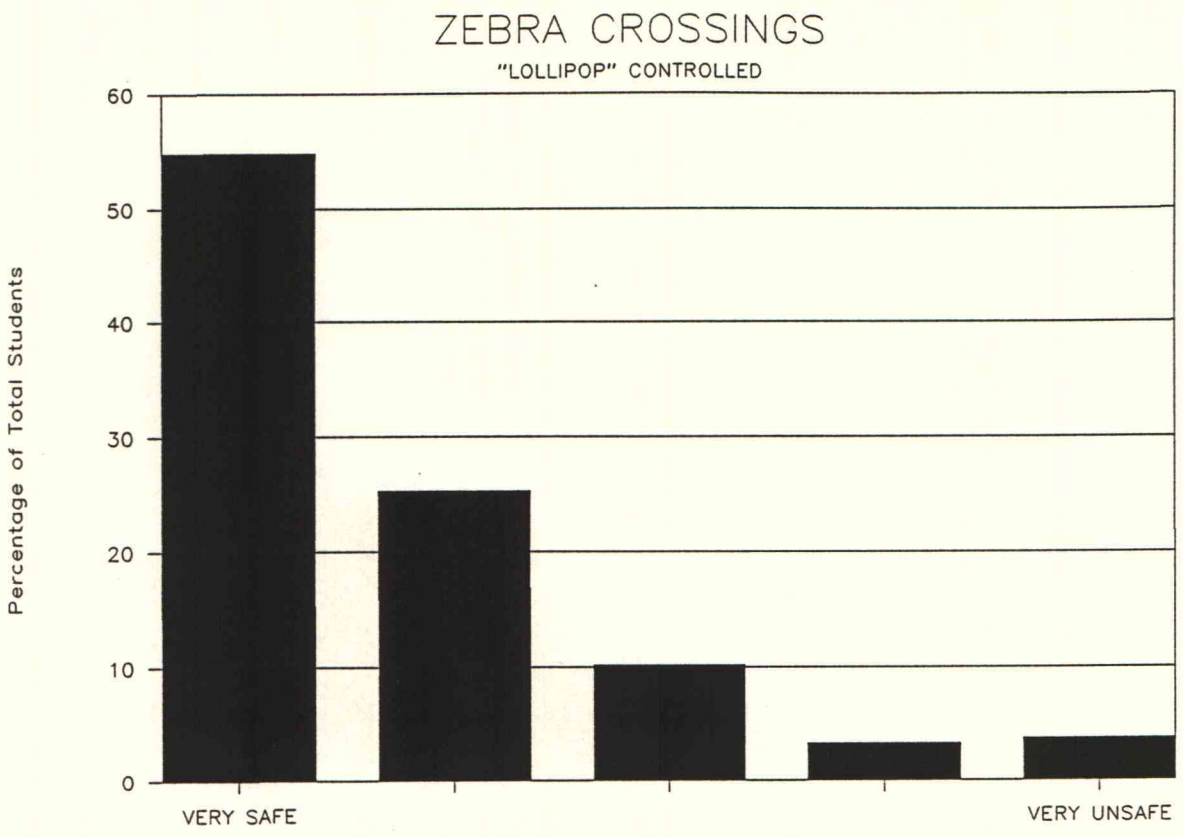


FIG 5.4 - Total Student Response

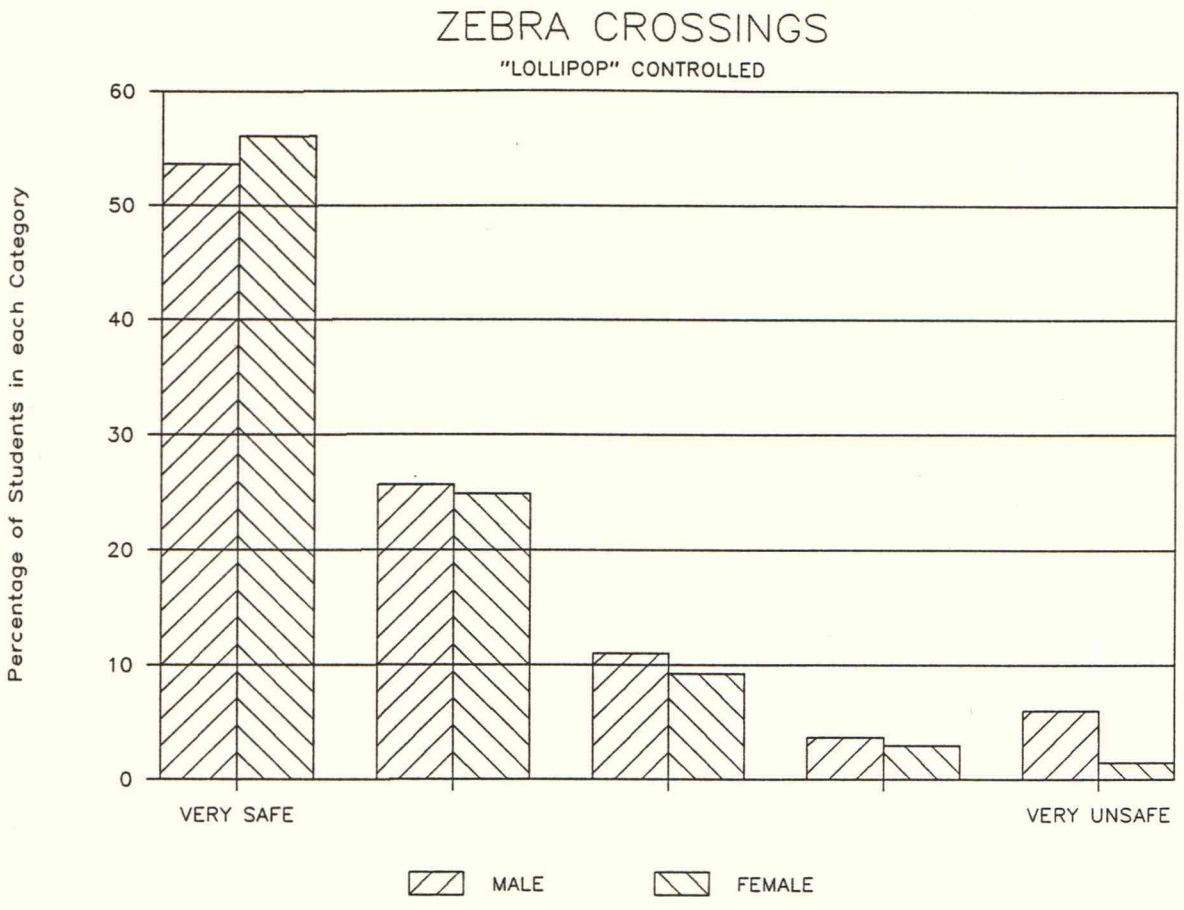


FIG 5.5 - Male/Female Student Response

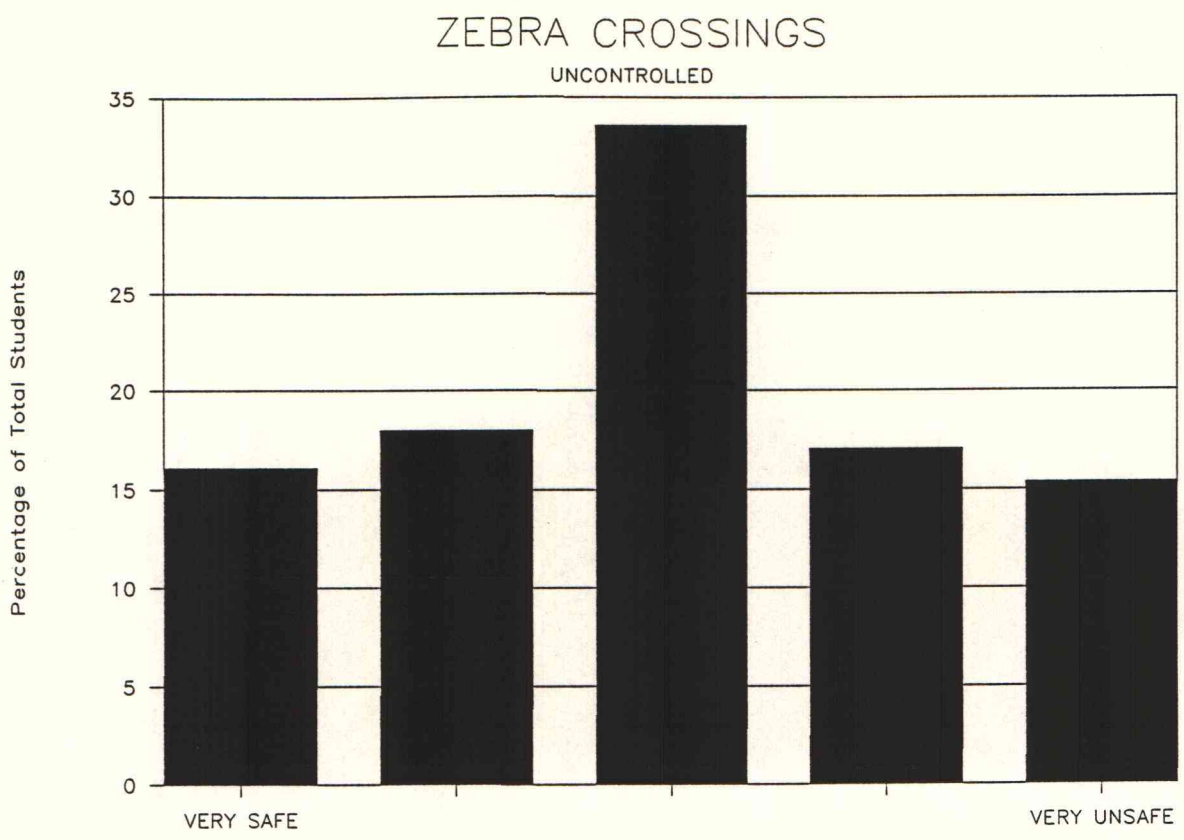


FIG 5.6 - Total Student Student Response

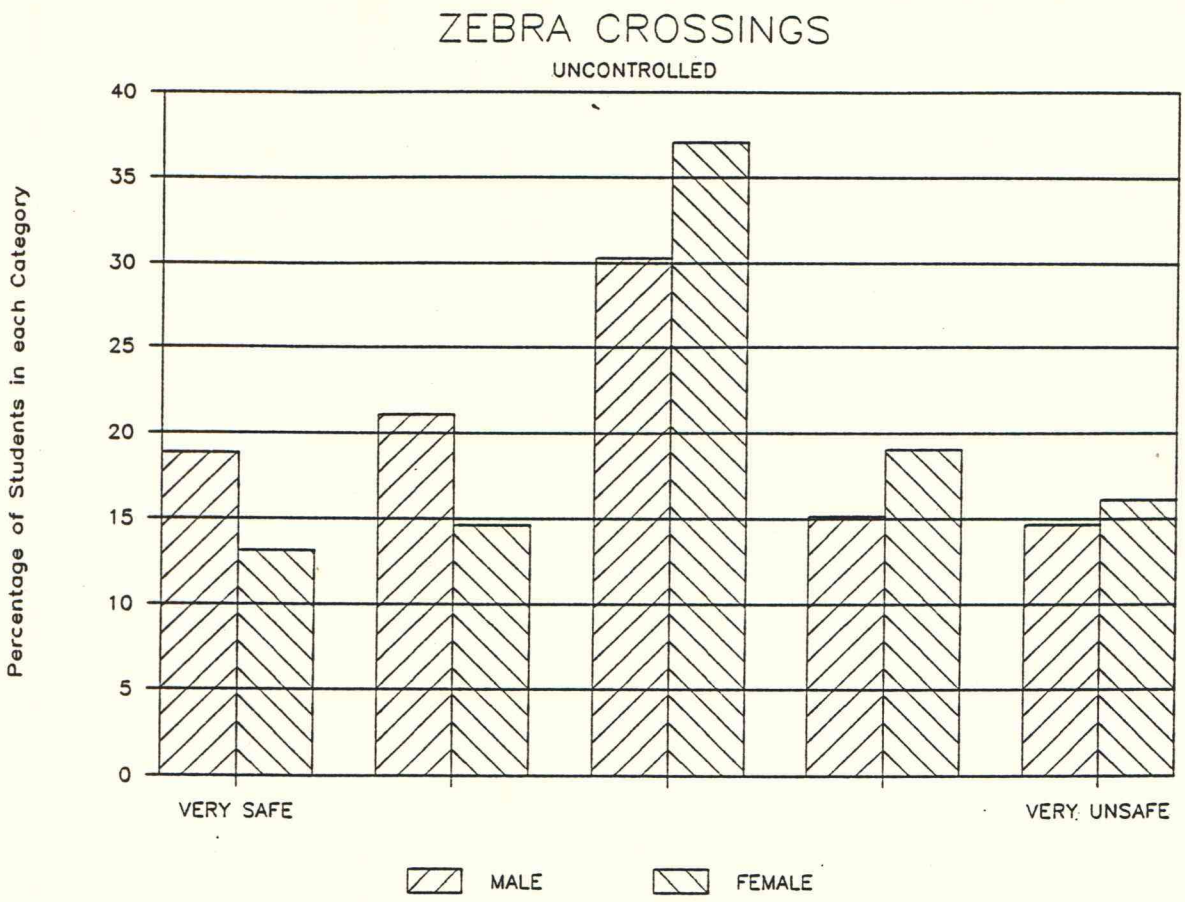


FIG 5.7 - Male/Female Student Response

YEAR LEVEL	MALE	FEMALE	TOTAL
4	9.6	5.3	6.8
5	11.5	2.3	7.2
6	12.9	3.6	8.5
7	17.8	0.0	11.6
8	0.0	15.0	7.6
9	20.0	0.0	8.5
10	2.9	10.0	5.5
11	13.3	21.4	17.2
12	17.4	3.7	10.0
TOTAL	12.8	6.6	9.9

FIG 5.8 - Students (%) involved in Pedestrian Accidents

more safer than uncontrolled intersections. Males on the whole find uncontrolled crossings more safer than girls (Fig 5.7), and they both tend to agree on the safety of "lollipop person" controlled crossings (Fig 5.5). It is interesting to note the percentage of males who find "lollipop controlled" crossings very unsafe. The majority of students who had been involved in accidents answered "very unsafe" here, and for the uncontrolled crossings, which makes it likely that these children were struck on these type of crossings. Quite e few students answered "very unsafe" for "lollipop controlled" crossings, and "very safe" for uncontrolled, which leads me to think they misunderstood the question. All the year levels followed the same pattern as in Figs 5.4 and 5.6, with the exception of the year 7 curve for uncontrolled crossings. The year sevens considered these crossings in the majority as "very unsafe", whereas every other year level finds them of average safety.

Question four was used to determine the number of students who had been involved in pedestrian accidents and the percentage that went unreported. Fig 5.8 gives the break down of students that have been involved in a pedestrian accidents. Overall 9.9% of the students have been involved, and only 4.3% were reported. It is quite a sizeable figure, (one in every ten roughly) and shows that children are in the high risk bracket when it comes to pedestrian safety. It also shows that a lot more pedestrian accidents are not reported than the number that are, indicating the number of accidents that occurred in Rockhampton could be up to twice as many as the 82 reported. It is quite possible that some of the students may have included bicycle accidents in their answers but it can only be judged on the information given. Males are involved in more accidents than

females and year 11 has the highest percentage of students involved, followed by year 7. Ironically these two age groups found it easiest to negotiate roundabouts in question two. Added to this fact is that year 11 found uncontrolled crossings the safest of all the groups. So perhaps it may be a case of these two year levels just underestimating the care and attention required to walk safely around the street network.

5.3 Adult Survey

This survey was conducted in the City Heart Mall, and the Rockhampton Shopping Fair on a question/answer basis. People were approached at random with the emphasis on obtaining a wide cross section of ages and an even balance of males and females. A total of 317 people were surveyed, with very few people approached refusing to take part in the study. The interviews were done over two days with each interview taking approximately two minutes including introductions and explanations.

Overall, 161 males were studied, and 156 females studied (some literally more than others!) in the survey.

The questions asked were very similar to the questions asked to the students, with the only real difference being that the safety of signalled crossings and uncontrolled zebra crossings were compared in question three, and that question one was dropped.

Fig 5.9 shows the results of the question pertaining to roundabouts. As with the student survey over half of the people surveyed found some difficulty with roundabouts. Once again males tended to find it easier than females and it was noted that the majority of people who expressed difficulties were elderly.

Figs 5.10 and 5.11 are the summary of the personal feeling of safety at the two crossing types. As can be seen 80% of people feel very safe when using the traffic signal crossings and this is what would be expected. Not a single person felt that these crossings were of less than "average" safety. The trend once again is for the males to feel safer than the females, and this generally goes for uncontrolled crossings as well. The bars tend to exhibit the same pattern as for the students, peaking in the centre. The elder female persons strongly felt that they were in danger when using these crossings in general.

The percentage of people involved in pedestrian accidents was a lot lower than that determined in the student survey. The males were higher than the females with percentages of 1.9 and 0.6 respectively. Half (2) of these accidents were reported to the police.

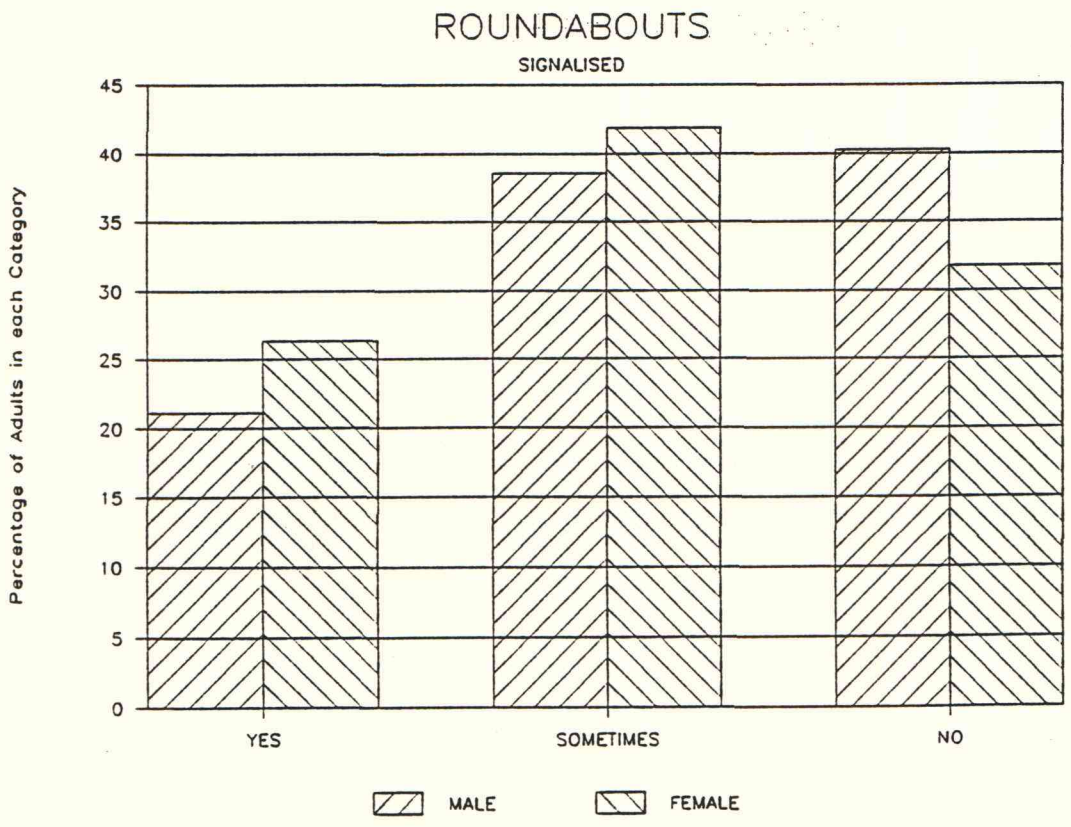


FIG 5.9

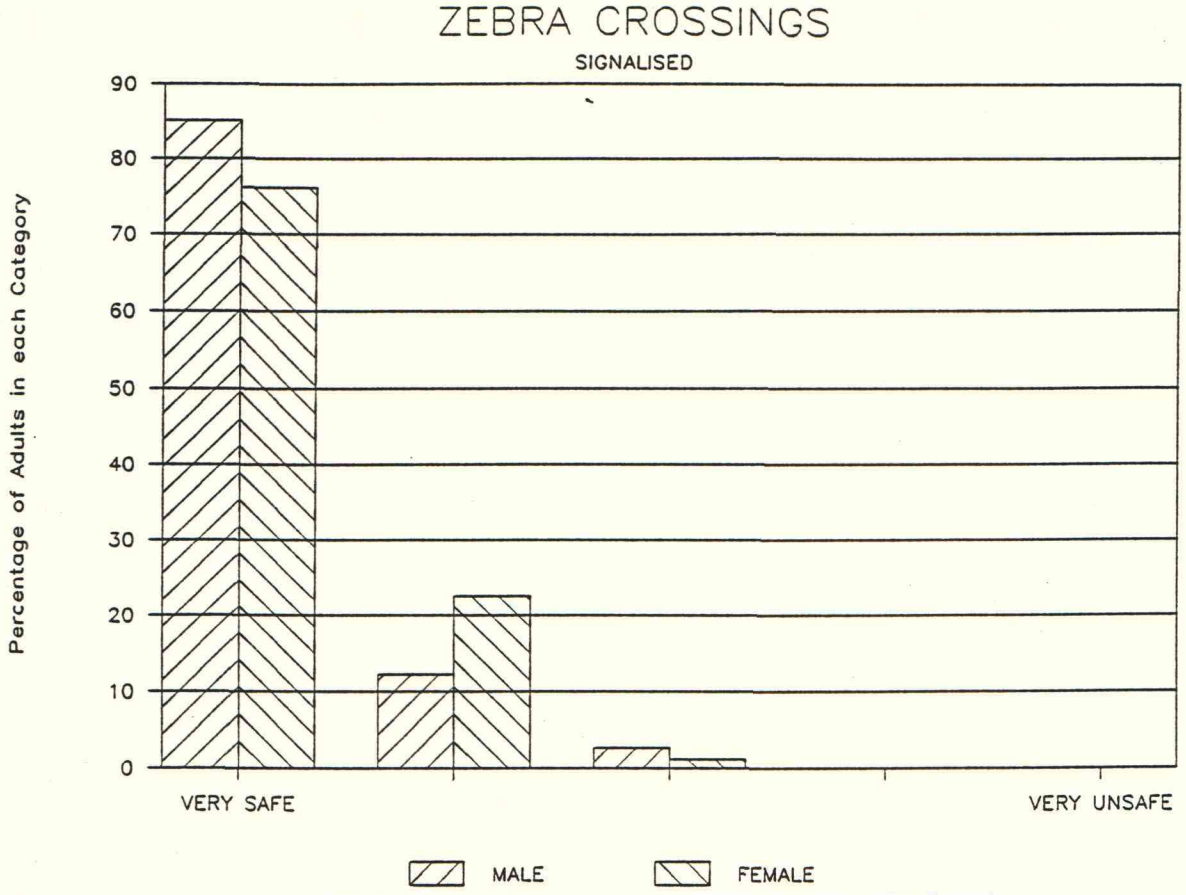


FIG 5.10 (above)

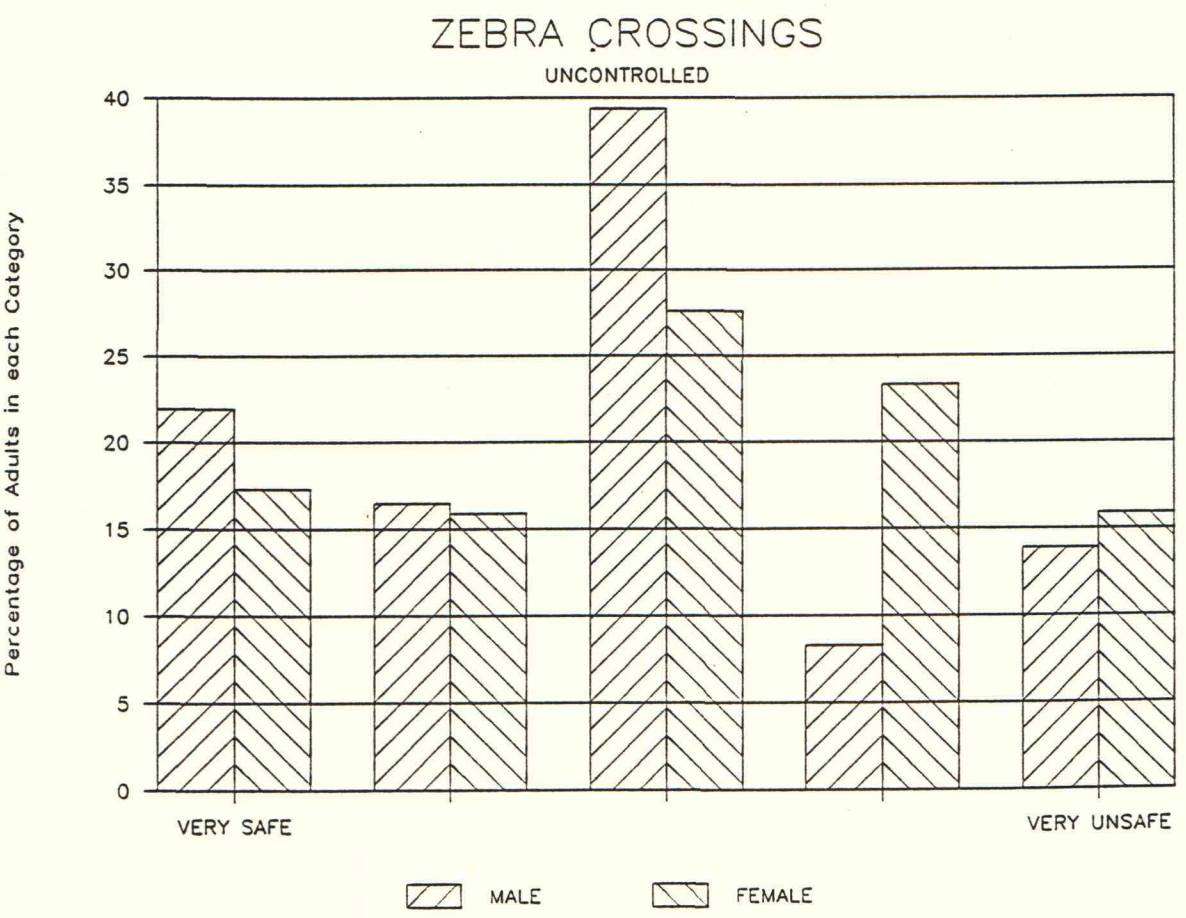


FIG 5.11 (below)

5.4 Bridge Surveys

The surveys on the bridge were conducted to gauge public opinion on the following:

1. Safety of the bridges.
2. An extension of the barrier on the Neville Hewitt Bridge.

The surveys took three days to complete with two days spent on the Fitzroy Bridge (one on either side) and one on the Neville Hewitt Bridge.

73 people were surveyed on the eastern side of the Fitzroy Bridge, 52 on the western side, and 63 on the Neville Hewitt Bridge. The safety scale used in the student and adult surveys was used again to obtain a measure of the pedestrian's feeling of safety. Fig 5.12 shows a stark contrast in the two bridges. The Fitzroy Bridge did not receive one "very safe" vote whereas The Neville Hewitt Bridge was considered to be very safe almost unanimously by the respondants. The difference in the bridges is that the N.H. Bridge takes pedestrians down one side only, and has a one metre high concrete barrier between the pedestrian lane and traffic lanes, whereas the Fitzroy Bridge has only a level seperation, and pedestrians are taken down both sides. A lot of the respondants commented that it is "only a matter of time" before someone is killed walking along the Fitzroy Bridge, and 13 of the people stated they would not use the bridge by choice, but they had no alternative. Bridge widening, taking bicycles down one side and pedestrians the other, barrier seperators were the most common remedies suggested by the people.

The other part of the survey involved questioning pedestrians of their destinations at the northern end of the

Neville Hewitt Bridge. Presently pedestrians are taken down the side of the approach embankments of the bridge, onto Glenmore Road. The concrete barrier and pedestrian lane ends here however the shoulder width of the bridge is constant as it continues on down to the natural surface. The outer concrete barrier also continues to natural surface. Pedestrians wishing to continue walking down the highway at present have to get over the concrete barrier and then walk for a fair distance on the shoulder beside the traffic.

Of the 63 people I questioned 41 said it would be much more convenient to continue along the bridge and down the highway, rather than drop down onto Glenmore Road and take a longer journey. 33 of those 41 people actually did negotiate the barrier and continue their trip. From that 33, 27 felt they were not safe at all whilst still on the bridge and in conflict with the traffic. The speed envrinment in this particular region was 80 km/h and this compounded the risk to the pedestrian. Two accidents had occurred in this section in 1989 resulting in death to one of the victims.

DEGREE OF SAFETY	NEVILLE HEWITT BRIDGE	FITZROY RIVER BRIDGE
VERY SAFE	60	0
-	3	11
-	0	36
-	0	29
VERY UNSAFE	0	49

FIG 5.12 - Safety Comparisons

6 COMPUTER DATA HANDLING/ANALYSIS

6.1 Hardware/Software

The computer system designed as part of this project has been implemented in Turbo Pascal V5.5. All that is required to run the system is one disk which will include the source code as well as the compiled executable code. It is not necessary that a copy of Turbo Pascal also be supplied.

The hardware required to run the system is any Personal Computer with a 5.25 inch disk drive. All input will be through the keyboard. Output will be displayed on the screen. It is proposed that the computer used for running the system have at least a 286 12Mhz processor otherwise the running of the program will be too slow.

The program can be run by typing PEDAC at the A> prompt however this will be done within the AUTOEXEC.BAT file so the user will be confronted with the main menu before having to touch a key. The menu requires the use of the function keys and is self explanatory.

PEDAC.PAS is the main program file however it also calls three other units which contain procedures to be used throughout the system. The program was too large for turbo pascal to handle in the one program. The total number of lines comes to 3700.

6.2 The Problem

The system was modelled on the procedure used by the Queensland Police Department to record details in the event of an accident. The procedure involved filling out Form P.T. 51 8/89 - Traffic Accident Report Form. When the form is completed it is filed away with all other files in large cabinets in the

filing room at Regional Police Headquarters. Such a procedure has three major problems which this system will rectify:

1. The storage of data takes up a large volume of room. With every police file, be it Accident Forms, Assault Charge Forms or Break/Enter Records, stored in together using a common filing reference number system, it is time consuming and difficult to retrieve data when required. The system is difficult to maintain and is an obvious risk in the event of fire.
2. The actual procedure of filling out the forms is time consuming when the members of our police force have more important things to do with their time. The actual forms are very congested and in great detail. All accidents are recorded on the same standard form irrespective of whether they are pedestrian accidents, bicycle accidents or vehicular accidents and there are many fields which apply to only one particular type of accident.
3. Once the information is stored away it becomes redundant and is very seldomly used. There is no means by which the data can be correlated for research purposes in determining the cause of accidents or determining any patterns which occur throughout the city.

Through computerising the procedure these problems may be overcome. For the purposes of this project the system is specifically designed to cater for pedestrian accidents only.

6.3 Purpose of the System

The idea for the system originated when I approached the Rockhampton Police to obtain data to research accidents involving pedestrians in Rockhampton from 1985-89. After quite a

few discussions and meetings with various personnel I was allowed to go through the filing system, after hours, with one senior police officer.

This officer was transferred shortly after I commenced searching, and once again I had to go through the channels to obtain permission to get the data. This time however I was denied access to the files, the reason being that confidentiality was being breached in that names were on the forms, and because all police files were kept together, it was not possible for me to sift through these files.

It was then brought to my attention that I should never have been given access to the files at all and that the original authorization was not officially given. From the time I did spend in the filing room it was estimated that it would take 29 hours of continuous searching to retrieve approximately 82 files.

I realised then that with a good efficient computer system, the information required could have been supplied with the push of a button and within five minutes, completely free of names and thus retaining confidentiality.

6.4 System Achievements

The major objective of the system is to be able to extract data for any purpose which the police may require. This task is quite complex. Overall there are more than 30 fields of data which are stored concerning each reported accident. The police may wish to retrieve data by setting criteria on any of these fields. eg. List all accidents occurring at intersections where the pedestrian was drunk and under the age of 18.

Being able to retrieve information in this manner is the

major goal of the project.

The input of accident data is prompted by a standard input screen of which there will be two pages. Each field name will be highlighted when a field value needs to be entered. There are two different means by which the user will input data. The first is via pop-up windows which gives the user a choice of pre-empted values. The other is by simply typing the given value. An example of this is age - the value is numeric and must be typed through the keyboard.

Viewing accident data requires the user to specify which accident number is to be viewed. The accident data is displayed in two forms identical to those used for inputting the data.

The system must provide an option such that the user can list the accident numbers on the screen.

6.5 System Method

The system was initially started using Dbase IV however being a higher level language, it was foreseeable that some tasks may become a problem, hence the use of Turbo Pascal V5.5.

The data is stored in a record of numeric, real, and alphanumeric arrays. At run time a tree is created which contains the record number and is indexed by the Accident Number. This tree is used to prevent timely searches of the data file when a specific accident needs to be retrieved. The program only needs to search the index tree for the accident number which provides the system with the appropriate record number to be retrieved. Such a method provides fast access to the system data.

Problems arise however when an accident record needs to be retrieved using any other field as the key. The data file must

be searched linearly looking at each particular record which can be time consuming.

The MENU procedure in one of the units is used extensively throughout the program for windowing and providing menus. Many of the data fields have a set selection of values which the user may choose. These selections are displayed in windows by the MENU procedure and are stored as numeric values. The set of selections for each field and the position of the window are passed to the procedure. It then returns a numeric value to represent the value chosen. The menus provide an inverse video display and shadowing and the user may select an option using the Arrow and Return keys.

6.6 Data Flow Diagrams

The following figures show a detailed description of the systems data and how the data interacts with the system. Overall there is very little interaction between the program and the user by way of input. The program is designed with an emphasis on data retrieval and is very "user friendly".

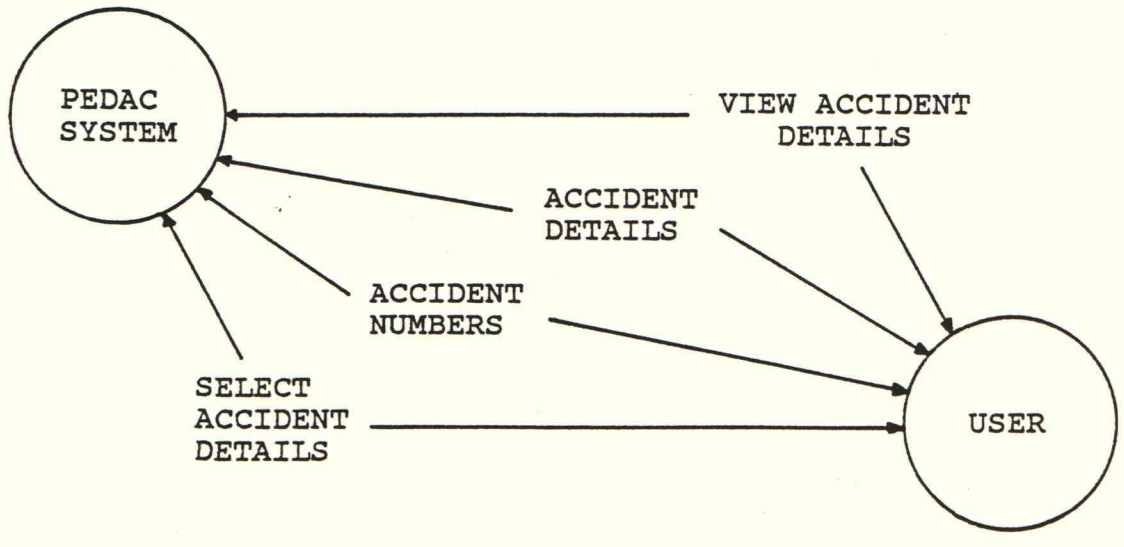


FIG 6.1 - Context Diagram

FIG 6.2 - Diagram 0

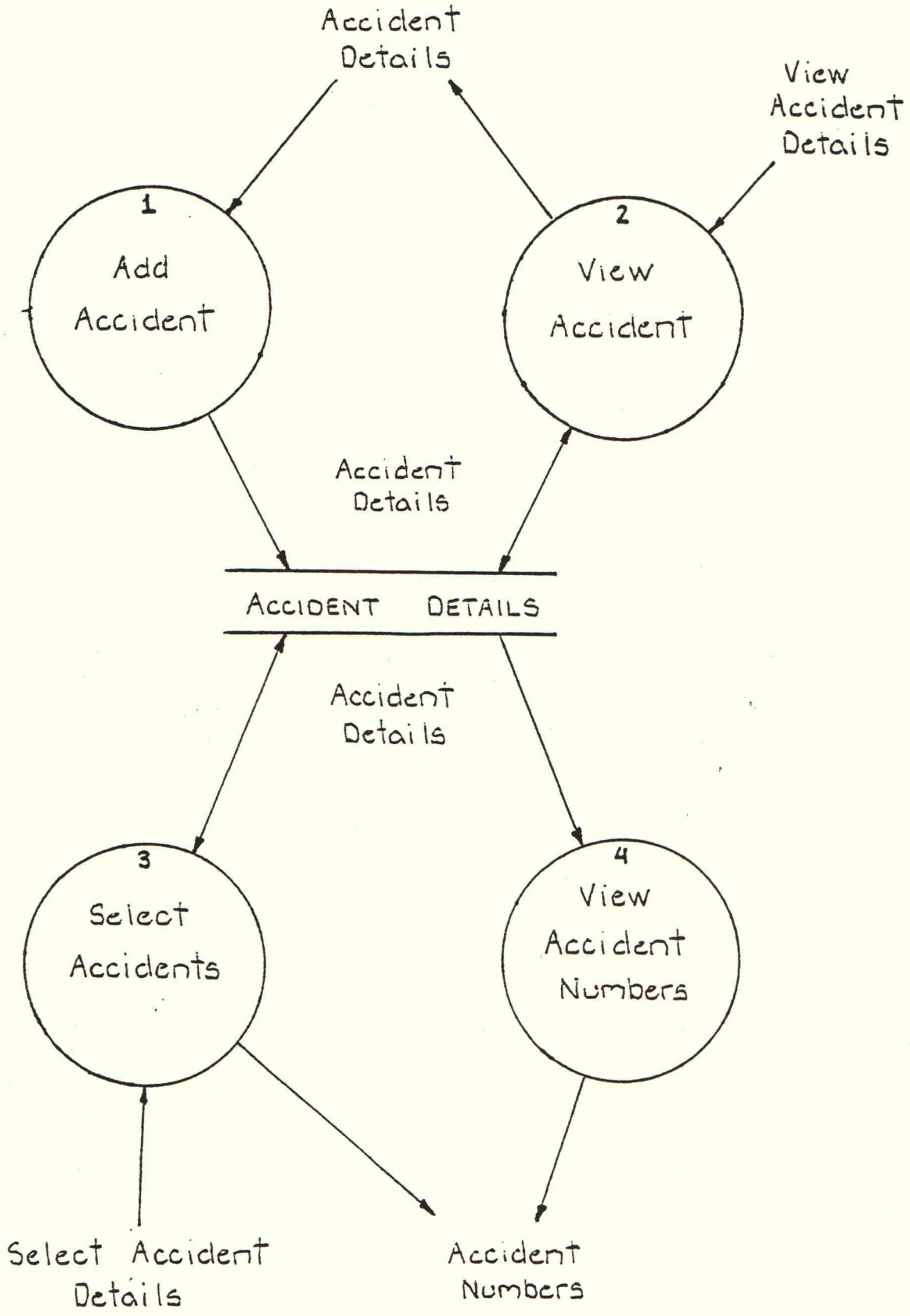
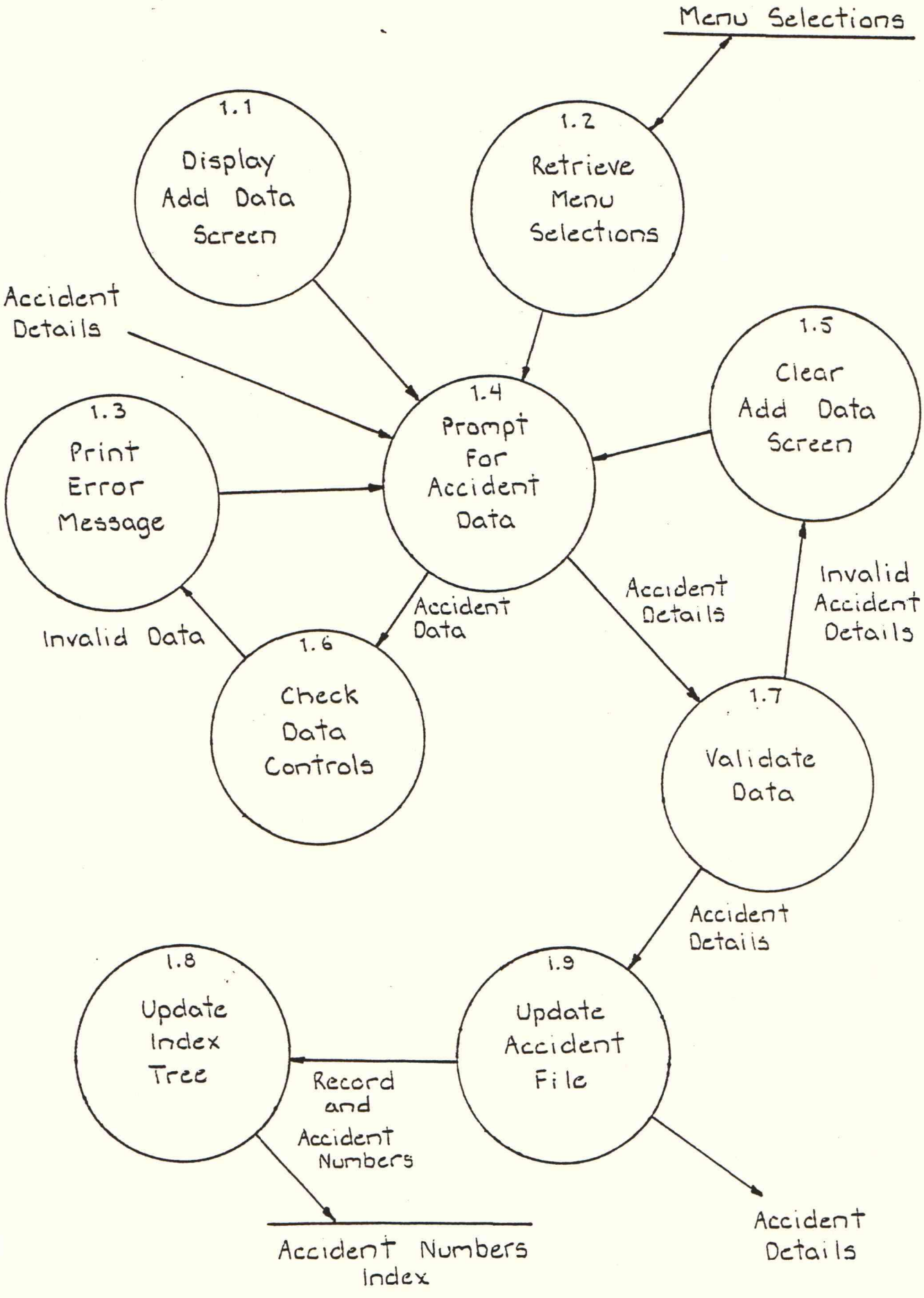
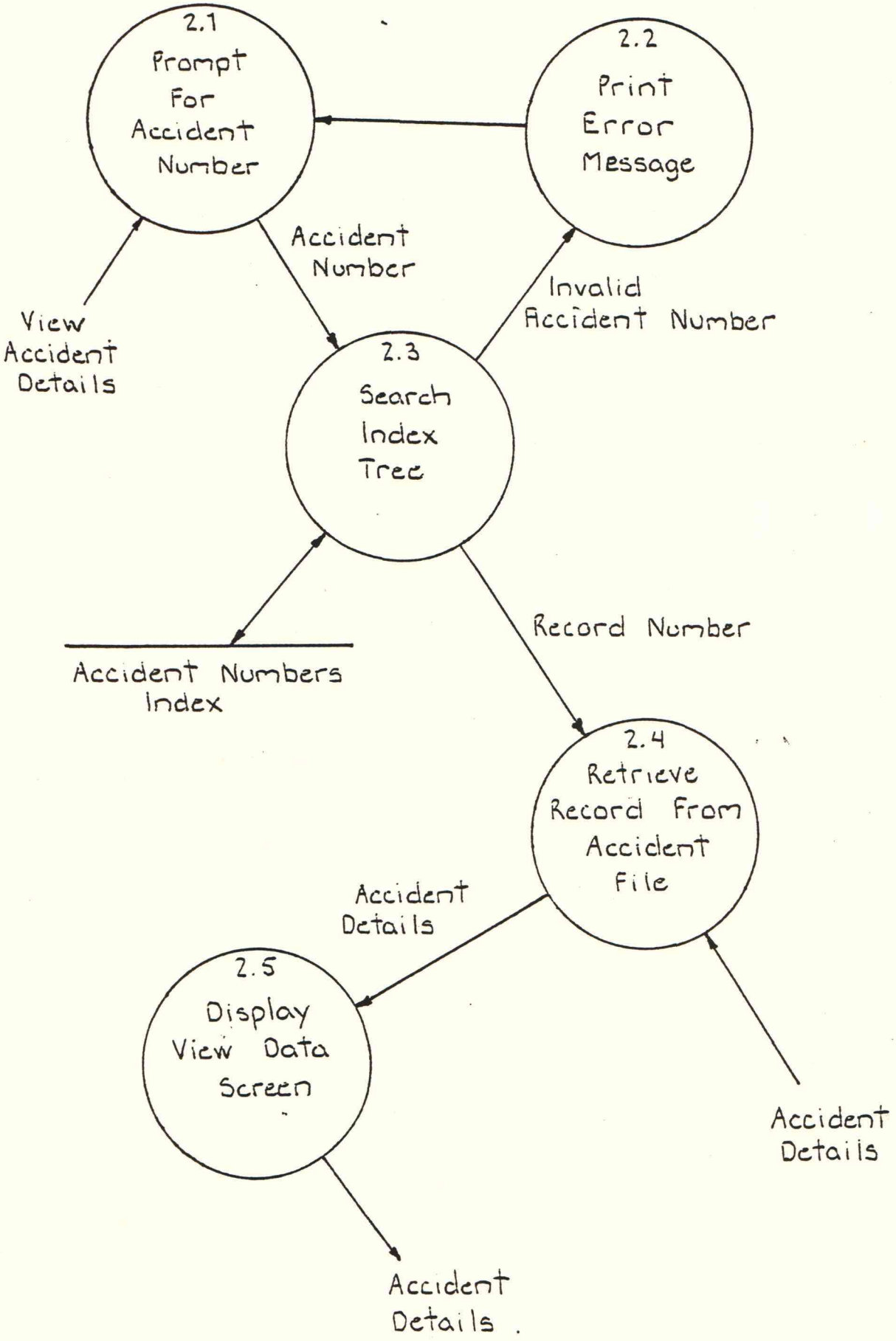
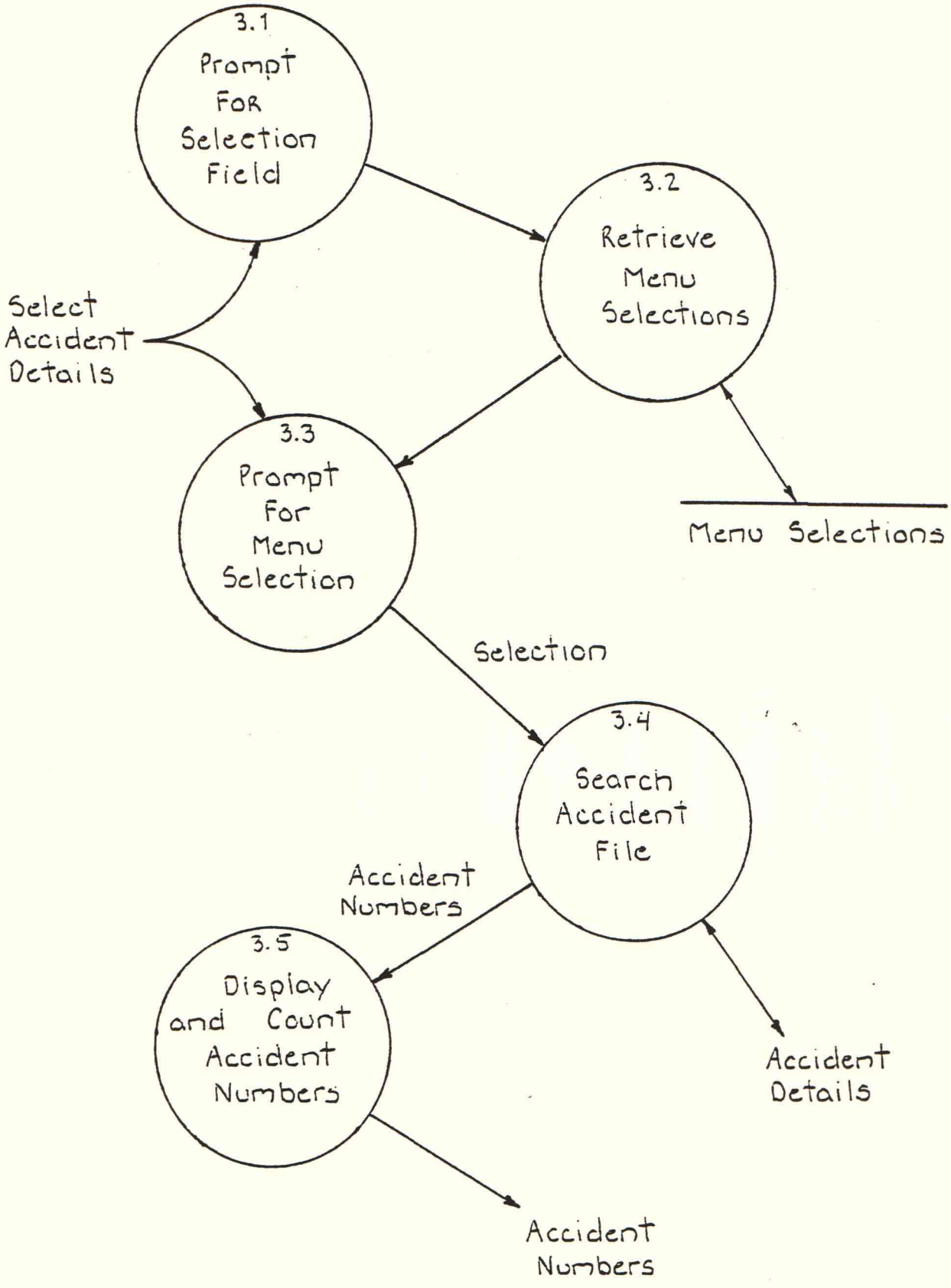
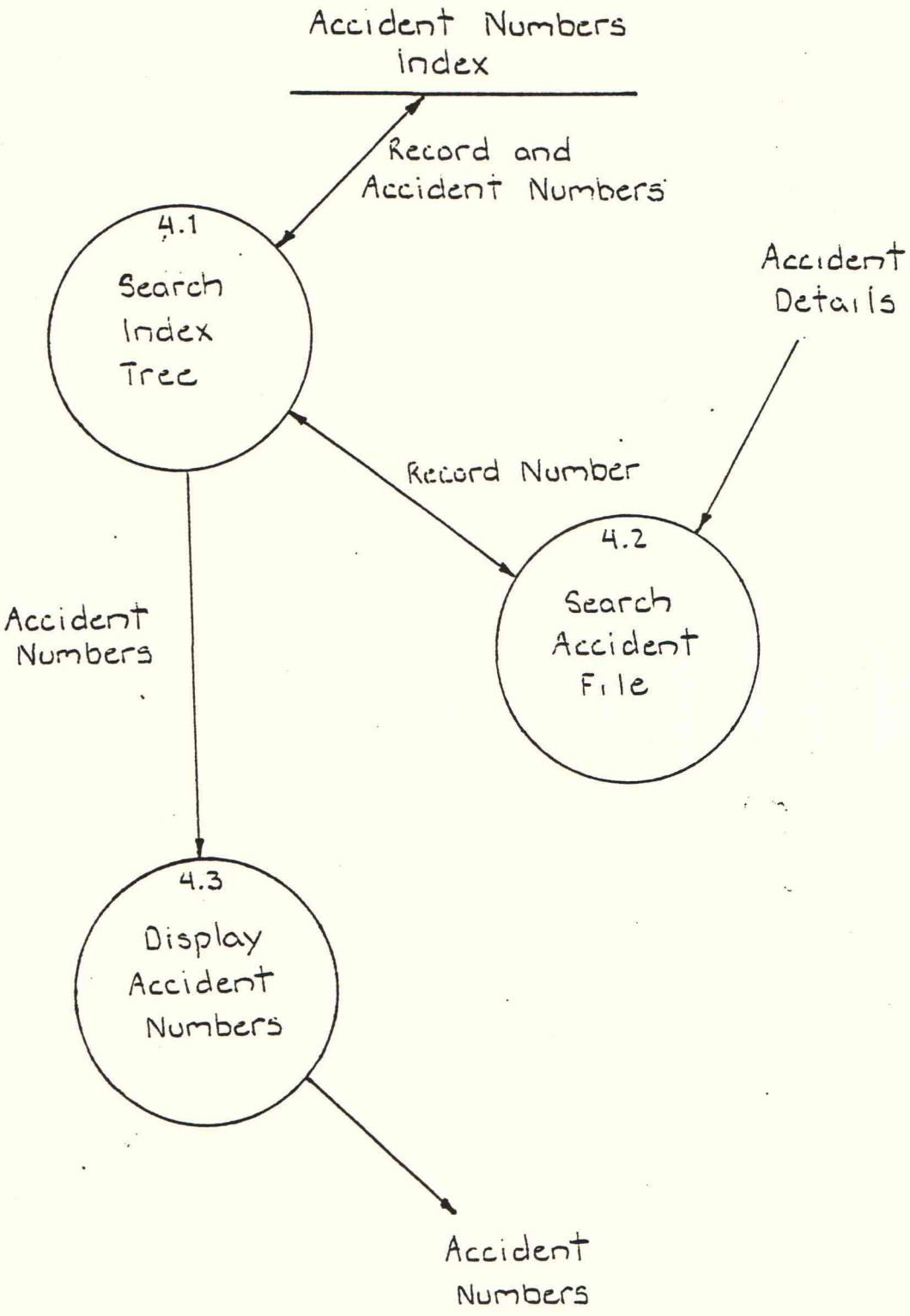


FIG 6.3 - Lower Level Diagrams




2.1
Prompt For Accident Number
2.2
Print Error Message
Accident Number
Invalid Accident Number
View Accident Details
2.3
Search Index Tree
Accident Numbers Index
Record Number
2.4
Retrieve Record From Accident File
Accident Details
2.5
Display View Data Screen
Accident Details
Accident Details


3.1
Prompt For Selection Field
3.2
Retrieve Menu Selections
Select Accident Details
3.3
Prompt For Menu Selection
Menu Selections
Selection
3.4
Search Accident File
Accident Numbers
3.5
Display and Count Accident Numbers
Accident Details
Accident Numbers


Accident Numbers Index
Record and Accident Numbers
4.1
Search Index Tree
Accident Details
Record Number
4.2
Search Accident File
Accident Numbers
4.3
Display Accident Numbers
Accident Numbers

6.7 Structure Charts

Fig 6.4 below is a structure chart of the main program of the Pedestrian Accident (PEDAC) System. It shows the modules to be called for each of the options available for the system and is similar to Fig 6.2 in the previous section.

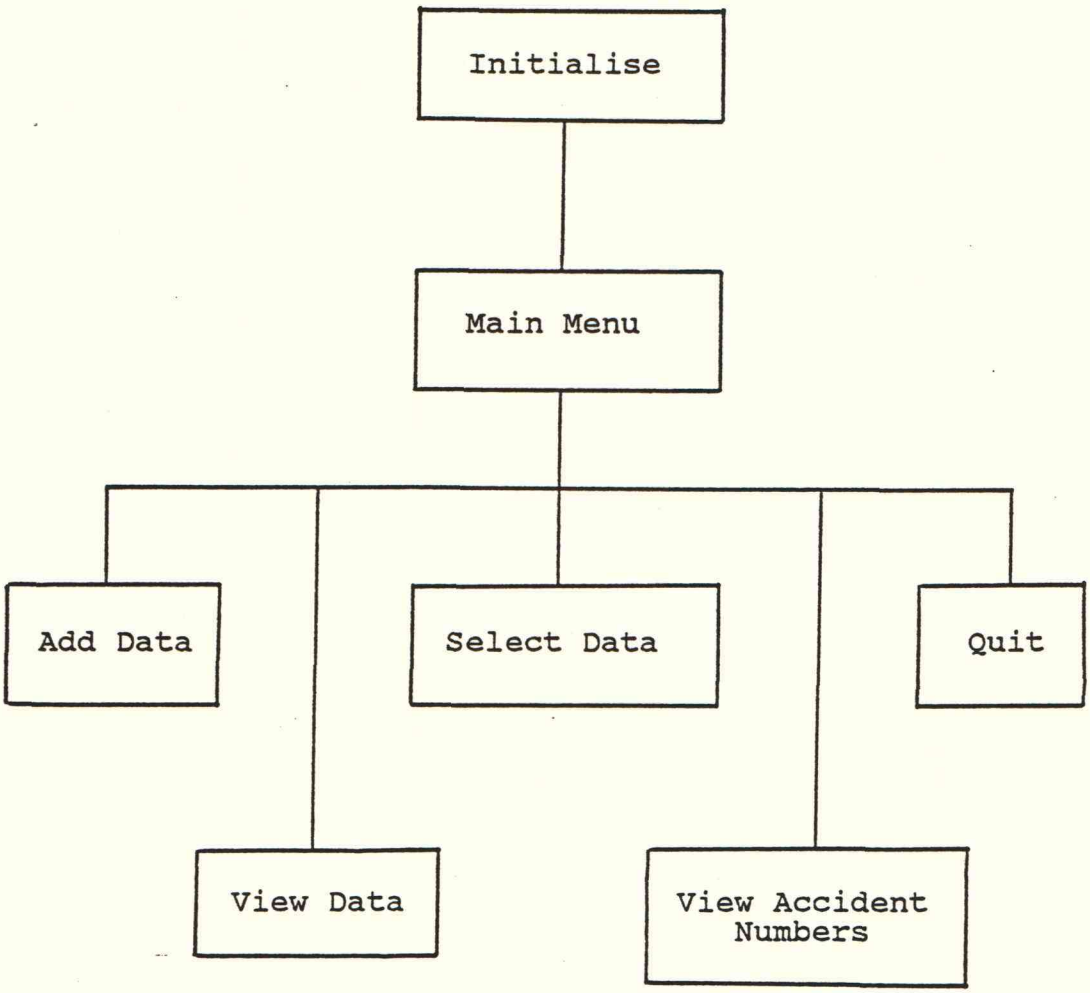


FIG 6.4 - PEDAC Structure Chart

Each of these windows can be broken down into a number of lower level structure charts showing the procedures and modularisation of the system. This is however simply another way of representing the processes shown in Fig 6.3. These diagrams are sufficient enough to provide a specification of the module design.

6.8 Data File Design

The system as given derives its data entities from the Accident Report Form. The entities which are relevent to pedestrian accidents are given below.

Accident Number	Numeric
Date of Accident	Numeric
Day of Accident	Numeric
Hour of Accident	Numeric
Contributing Circumstances	Numeric
Light	Numeric
Atmospheric Conditions	Numeric
Road Surface	Numeric
Horizontal Features	Numeric
Vertical Features	Numeric
Traffic Control	Numeric
Special Features	Numeric
Divided Road	Numeric
Speed Limit	Numeric
Unit Involved	Numeric
Pedestrian Age	Numeric
Pedestrian Sex	Numeric
Pedestrian Drinking	Numeric
Vehicles Intention	Numeric
Severity of Injuries	Numeric
Driver Age	Numeric
Driver Sex	Numeric
Driver Drinking	Numeric
Drivers Licence	Numeric
Road User	Numeric
Driver B.A.C.	Real
Road Name	Alphanumeric
Intersecting Road	Alphanumeric

In accordance with the Accident Report Form there occurs one and one only value for each of these variables for each accident. Some of the fields may be left unanswered on the form such as "Intersecting Road" and "Driver B.A.C.". There are no repeating groups.

The data entities may be broken down into different categories such as Pedestrian, Driver, Road, Accident. An example of such relation is:

Driver = {Accident Number + Driver Age + Driver Sex + Driver Drinking + Driver B.A.C.}

The three other relations are formed similar to this set. One visible feature of the relations formed which will be consistent with the other three relations is that the Accident Number will be the key to the table. Therefore using such methods as Fact Base Analysis or Normalisation the recommended data structure will be a single table with the accident number as the key. This form is consistent with the layout established in the Accident Report Form.

Turbo Pascal provides for such a file structure through the extensive use of records and arrays. The structure recommended to cater for the numerous data types is as given below.

```
Type
     String30 = String[30];
     DataType = Record
                    Numeric : Array[1..26] of LongInt;
                    Reals : Array[27] of Real;
                    Strings : Array[28..39] of String30;
                End;

Var
     Data : Datatype;
```

The storage of such file will be external on disk. It is anticipated that the volume of data eventually stored on the master file will be too large for it to be loaded into internal memory.

The file will be initially indexed on the Accident Number. At run time the accident numbers will be read into a binary tree and will indicate the location of the corresponding record on the disk. This binary tree is loaded into internal memory and assists in direct searching of the data file by Accident Number. In the future more such indexes wll be required on other fields within the system.

If the file is to be searched by any field other than the Accident Number it will be done sequentially. Such a search will

be relatively slow especially as the volume of the data increases. For this reason it is suggested that further indexes be created on other popular selected searching fields within the system. Such indexes will also be stored externally and will be automatically updated whenever a new record is added to the masterfile.

6.9 Screen Design

The terminal serves four functions in the PEDAC system: Data Collection, Validation, Reporting, and Inquiry. Clear, concise, and accessible screen design not only helps users enter data accurately, but also creates an atmosphere of friendliness and ease in which users can feel comfortable and can more easily master the system. All screens within the system are based on a standard 24 by 80 format.

The following is the screen hierarchy the system uses:

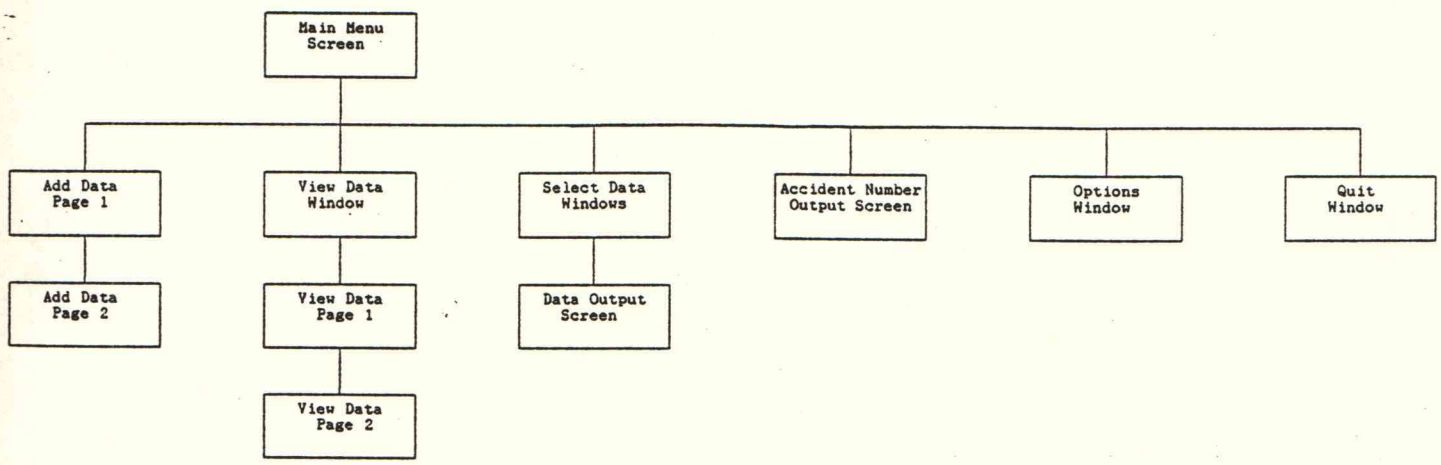


FIG 6.5 - Screen hierarchy for PEDAC

In all screens error reporting is handled by a window displaying a message as to what the error was and the next course of action.

6.9.1 Data Entry Screens

These screens require the user to fill in the blanks as highlighted on the screen in inverse video. By interfacing in this manner the user is less prone to confusion and makes the system more user friendly. On fields which provide a set selection of choices, they are displayed in a pop-up menu, which allows the user to scroll through them by using the Arrow keys. The selections are also highlighted in inverse video. The user may make a selection by pressing the ENTER key. The size of the field is shown and the field name is used to prompt the user for input.

6.9.2 Enquiry Screens

Enquiries are made through the use of windows. A window appears on the screen asking the user to enter some information. The windows are consistent across the screens and are easy to use. Pop-up windows identical to those used for adding data are also used to select the criteria on any field to be used to retrieve records from the data file.

6.9.3 Reporting Screens

These screens are in exactly the same format as the data entry screens in 6.9.1. Each field however contains the given value from the selected record. The other two output screens are in the same format by placing the relevant Accident Numbers in a window in the centre of the screen.

6.10 Menus

The use of menus in the system is limited. The main menu is

loaded at the start of the program. It involves the use of function keys (eg F2,F3,...) for the user to select the required option. Pressing ESC at the start of any option will bring you back to the main menu.

Pop-up menus are used throughout the system to prompt the user for input. They are used extensively in adding data to the system and also when selecting a field on which to search the data file. These menus, as previously explained, provide the user with a list of his available options. He may then scroll through the menu by using the Arrow keys. Once the required selection is highlighted, the user may select it by pressing the ENTER key.

6.11 Testing

The program has been fully tested using test data that try to force the system to fail. This included entering incorrect data which may corrupt the data file if not handled in the correct procedure. Such tests were essential to guarantee that the actual controls and error handling on certain fields were triggered in the right situation. The program came through the final testing unscathed.

6.12 System for the Future

Directing output to the printer is one facility which is presently not available to the user. The system provides no hardcopy reports on accidents or the results from data retrieval searches.

At present the volume of data on which the system relies is around 50 records. In a real world environment this could be up to ten thousand or more. If this is the case a more complex

system of trees and indexes will need to be created as a linear search on a file of this magnitude will be relatively slow.

An extension of the PEDAC system could be made to cover all types of traffic accidents. Many data items will be required for all such accidents and there will be many which will be unique only to vehicular accidents as there is to pedestrian accidents. The system should provide for some distinction between each different type of traffic accident.

7 CONCLUSIONS

Sections 2 to 6 have studied in detail the numerous details with respect to pedestrian accidents, safety, and behaviour in Rockhampton. A number of conclusions can be drawn from this information and these are summarised below.

7.1 Where Rockhampton Stands

1. Rockhampton lies below the national average for pedestrian casualties per total road casualties.
2. In comparing with the provincial cities of Queensland, Rockhampton has casualty rates of 0.08 deaths and 1.31 injuries per thousand capita. There are five cities with higher rates and five with lower rates.
3. There is scope to improve the pedestrian safety in Rockhampton and lower these figures further.

7.2 Accident Characteristics

1. Rainy weather at night is the highest risk conditions to the pedestrian.
2. A pedestrian is more likely to be involved in an accident at night on the weekend than any other time.
3. Accidents are more likely to occur at intersections than anywhere else, and predominantly at crosses. Traffic lights reduce the danger to pedestrians, and confusion over right of way at sign posted intersections appears to be a major cause, of pedestrian accidents.
4. Although the chances of accidents occurring are higher at intersections, you are more likely to be killed if struck away from them.

```
program pedac;
{$M 65520,0,65500}

uses
    crt,udfs1,udfs2;
var
   key:char;
   finished:boolean;

begin
     initialize;
     mainscreen;
     repeat
           cursor(off);
           key:=readkey;
           if key=#0 then
           begin
           key:=readkey;
           case key of
                f1: dohelp(1);
                f2: add_data;
                f3: view_data;
                f4: select_data;
                f5: show_acc_numbers;
                f6: options;
                f10: quit;
           end;
           end;
     until finished;
end.
```

5. The majority of accidents are the fault of the pedestrian showing undue care and attention. 78% of the accidents involved the pedestrian entering onto the carriageway from the footpath.
6. Unlicensed drivers are proprtionally involved in more accidents than licenced drivers and nearly always are responsible for the collision.
7. 80% of pedestrian deaths in Rockhampton took place on declared roads (Main Roads), and a pedestrian is more likely to be killed if struck on a declared road against any other street. Two-thirds of the accidents result in victims being hospitalised if not killed. Pedestrians are responsible mostly for accidents causing death and hospitalisation.

7.3 Pedestrian Behaviour Observations

1. Women have a better level of pedestrian behaviour than men in Rockhampton.
2. Pedestrians tend to use signalised crossings better than uncontrolled zebra crossings in the city. The "lollipop controlled" school crossings are the best examples of pedestrian behaviour in the city.
3. There doesn't appear to be any real change in pedestrian behaviour from the Central Business District to the suburban pedestrianised areas.

7.4 Pedestrian Surveys

1. More primary school students walk to school overall than secondary students.
2. More than half of the school students find some difficulty in negotiating a roundabout on foot. Males in general find

it easier to cross than females.

3. Students regard "lollipop controlled" crossings a lot safer than uncontrolled crossings. Males have more confidence with uncontrolled crossings than females.
4. Quite a large number of students have been involved in unreported pedestrian accidents. Males are involved in more accidents than females on the whole, and year 11 and year 7 students have the highest involvement in accidents.
5. The majority of elderly people find difficulties at roundabouts, and feel insecure at zebra crossings about their safety. 80% of the people feel "very safe" when using signalised crossings, and males feel safer than females on zebra crossings.
6. People in general, consider the Neville Hewitt Bridge "very safe" for pedestrian transport, but feel the Fitzroy River Bridge is quite the opposite, and is a risk to the safety of the pedestrian.
7. The majority of pedestrians using the Neville Hewitt Bridge believe the concrete barrier and pedestrian lane should continue along the highway on the northern side.

7.5 Computer Data Handling/Analysis

1. Police have little or no access readily available to quickly analyse and retrieve data in Rockhampton. The present filing system for traffic accidents in particular is antiquated and could easily be replaced with an efficient computer system, making it far easier and simpler to detect and monitor problem areas.

8 RECOMMENDATIONS

The preceding chapters have examined a number of factors pertaining to pedestrian accidents, behaviour, and safety. A pedestrian accident is the result of conflict between the pedestrian, driver and vehicle, and the surrounding environment. In any one accident a deficiency in one of these factors, or a combination of all of them, is the cause of the accident. The surrounding environment problems can be rectified using engineering solutions and the pedestrian and driver factors are addressed by education and enforcement of existing traffic regulations.

8.1 Engineering Solutions

One of the major problems with installing roadway improvements is that pedestrian accidents are usually randomly scattered over large areas and seldomly concentrated at any one site. This highly random nature of pedestrian accidents presents a problem to pedestrian safety design in terms of wise expenditure.

As outlined in section 3.3.1 there are few locations along the main roads that could possibly be considered blackspots. Going on this evidence there is one (possibly three) sites where remedial action needs to be taken. This location is the corner of Fitzroy-Kent Streets, with possibly the northern approach to the Neville Hewitt Bridge, 180m north along George St from the George-Fitzroy Street intersection, and on Gladstone Road in the vicinity of the Wagon Wheel restaurant.

The aim of providing protection for pedestrians is to increase the potential safety for the pedestrian. Ideally a

total segregation of vehicular and pedestrian traffic virtually guarantees 100% safety for a pedestrian. Unfortunately this is just not feasible as this would require the construction of underpasses and overpasses, at every street crossing, a ridiculous situation in even the most pedestrianized areas imaginable! So we have to look at making the conflict points as safe as practically and economically possible.

The most common conflict area is the pedestrian crossing, be it controlled, or uncontrolled. The results of the survey (Section 5) showed that the public in general feel that their safety is at some risk when using uncontrolled zebra crossings. Unfortunately their are no uncontrolled zebra crossings on the declared roads in Rockhampton and as a result there is no guide to the percentage of accidents that occur on these crossings in the city. However it is likely that if the public feel they are at risk when using them they will shy away from them. The visual impact of the crossing determines the potential safety for the pedestrian. Advanced pavement markings and signing increases this potential. I believe the pedestrian series (Fig 8.1) of regulatory and warning signs should be changed in colour to fluorescent orange from the standard yellow. The fluorescent orange is much more eye catching than yellow, and this is important when peoples lives are at stake. The colour change would increase the number of drivers who actually see the signs

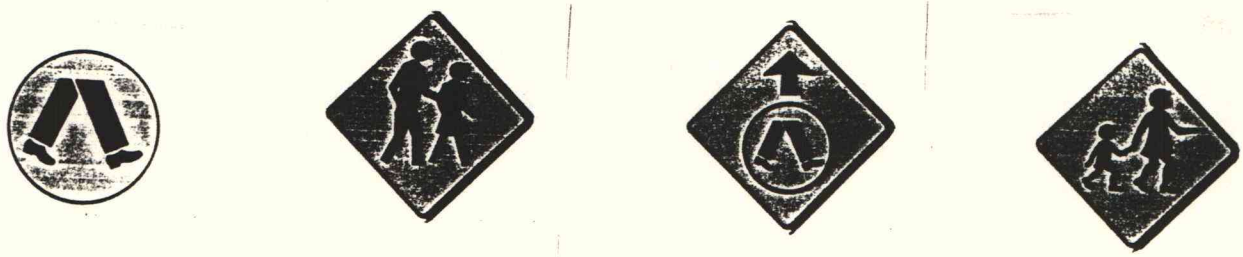


FIG 8.1 - Pedestrian Series of Regulatory and Warning Signs

and thus increase the potential safety of the pedestrian. The warning to drivers that pedestrians have legal right of way is important. At present a large number of drivers do not see the warning signs and as a result don't see the crossing until it is too late.

With the pedestrian in Rockhampton at a higher risk at night and in rainy weather, provision of good lighting is also important. Every crossing should be well lit. A Perth (Pegrum 1972) study showed the installation of special sodium vapour lighting reduced the number of night time pedestrian accidents by 60%. The majority of crossings in Rockhampton would not necessarily need this treatment but crossings in the C.B.D. and the busier crossings could be considered.

The introduction of reflectorized raised pavement markers (RRPM's) on the crossings to simulate the painted zebra striping is one treatment I recommend. RRPM's generally provide more effective and durable pavement markings than painted lines because they are not generally obscured at night under wet conditions, and they are conspicuous in all conditions. They are approximately $15 each to purchase and install and would no doubt be of great benefit to the pedestrian's safety, at a small cost (20 per crossing = $300). The placement of the markers is given in Fig 8.2.

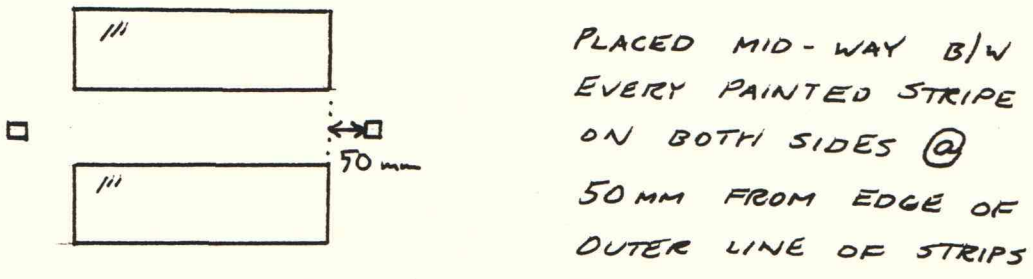


FIG 8.2 - Recommended placement of RRPM's

Median islands also are a means of increasing pedestrian safety. They provide an intermediate refuge for pedestrians without introducing undue interruption to vehicular movements. The pedestrian then only has to negotiate relatively narrow one way pavemants. This treatment is advisable for wide streets in busier areas. The use of medians in conjunction with zebra crossings has also proven to reduce the likelihood of accidents.

As to be expected people in Rockhampton feel very safe when using a signal controlled crossing. At present the pedestrian has to share his crossing phase with turning vehicles offering a continuous threat to human safety. At intersections of heavier flows (both vehicle and pedestrian) an exclusive WALK phase can reduce the hazard to pedestrians. The Fitzroy-East, Bolsover-Denham, and Fitzroy-Bolsover intersections are the most likely locations for this treatment.

Section 5 showed that more than half the people surveyed had some difficulty in negotiating roundabouts. Rockhampton has a number of roundabouts and with the installation of these devices increasing and the apparent difficulties associated with crossing them, special thought should be given to pedestrian movements in the design. Large roundabouts result in greater walking distances, however there are no "large" roundabouts as such in the city to consider. Zebra striping should not be painted on the pavement at the entrance and exit to roundabouts. Pedestrians should not be given a false sense of security, but be encouraged to identify acceptable gaps in the traffic. Splitter islands act as refuge islands for pedestrians at roundabouts. Where pedestrian volumes are high, or the proportion of young and elderly citizens are high, priority crossings may be considered. These crossings should not be

located within 20m of the exit to prevent blocking and choking of the roundabout. Fencing may be used to ensure the crossing is used correctly. For the Rockhampton roundabouts it is more desirable to install fencing and handrails on the splitter islands (providing width is adequate) to ease the difficulties associated with crossing. Most of the roundabouts will not even require any form of treatment.

The Fitzroy River Bridge is a particular area of concern with regards to pedestrian safety in Rockhampton. The majority of people interviewed expressed concerns for their safety when using the bridge and although there has been only one accident involving a pedestrian in the last five years there, the potential is there for more serious consequences. 21000 vehicles and 125 pedestrians use the bridge daily. The bridge has two grade seperated footpaths offering no protection to the pedestrian, and these paths also carry bicycles. This bicycle - pedestrian conflict takes place within one metre of the traffic as their are no shoulder widths on the traffic lanes. With these lanes being bounded by the kerb, there is the possibility of pedestrian or bike rider being forced or stumbling and tripping onto the carriageway.

This coupled with the high occurrence of vehicular accidents make the bridge a hazardous environment for the pedestrian. There are a number of possible solutions to the problem;

1. Use of concrete barriers.
2. Seperate pedestrians and cyclists (one either side). This could be achieved by using turnstiles at each end of the bridge to prevent cyclists from using the pedestrian side of the bridge.
3. Footpath widening.

All the above could be used in combination with each other as well as individually with money being the limiting factor.

The survey on the Neville Hewitt Bridge showed that the bridge is considered, and it is, very safe for pedestrians. However a number of pedestrians expressed that the barrier and lane should continue north down the bridge approaches until the outer barrier terminates, instead of taking pedestrians down onto Glenmore Road. Two accidents occurred in this location as the pedestrians do tend to negotiate the barrier and continue down the highway and the area needs attention. The barrier should be extended and a pedestrian pathway constructed away from the carriageway along the left hand side of the highway to the Knight St intersection. With the imminent development of Kershaw Gardens and the proximity of the Shopping Fair and K Mart plaza, the pedestrians can be taken across the road at the intersection by means of a pedestrian activated WALK signal and the path continue along the right side of the highway past the waterfall to the High St lights.

Improved lighting needs to be looked at at the George St (Section 3.3.1) site where the two accidents occurred. This area can be quite busy with the Mobil Coach Terminal and McDonalds located here. The shrubbery on the median island also needs to be kept constantly trimmed to a height of 0.5m to allow clear visibility to both driver and pedestrian. A paved surface with fencing on the median may be warranted in the future.

A Department of Transport pedestrian count showed a similar treatment is required on Moores Creek Rd adjacent to the Shopping Fair. It revealed that 142 people cross between the Fair and Kerr, Tynan Sts, and 268 cross at the traffic lights at teh Yaamba Rd - Moores Ck Rd intersection. (See Fig 8.3).

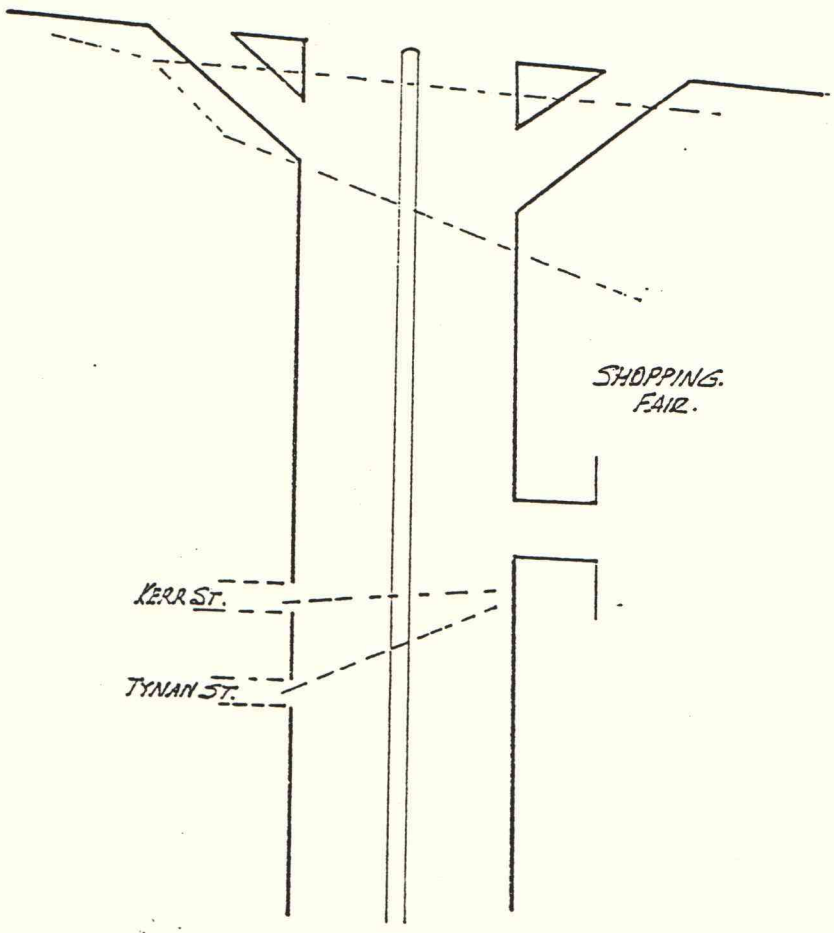


FIG 8.3 - Pedestrian Crossing Paths

Moores Creek Road (near Shopping Fair

Because of the 80km/h speed environment, a concreted median strip with handrails and fencing is recommended, and the shrubbery trimmed to 0.5m for 50m either side. Advanced warning signs should also be provide. A zebra crossing was not considered because the road is the National Highway.

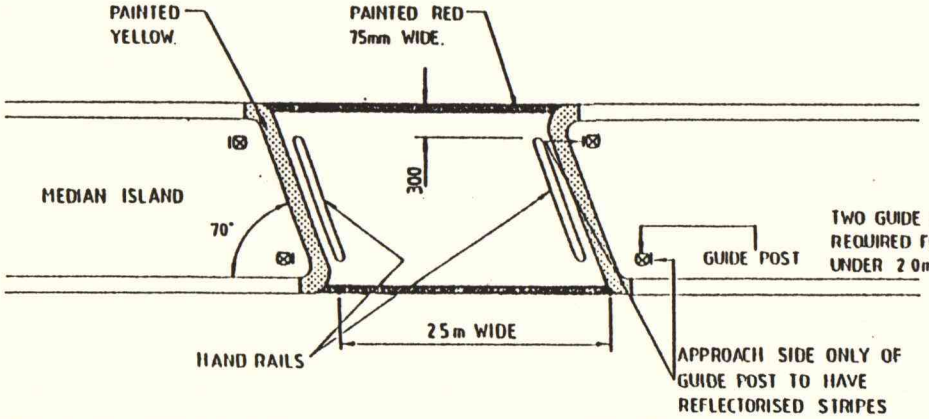
Provision for pedestrians should be made in the signal program and appropriate pavement marking and island treatment carried out, at the traffic lights.

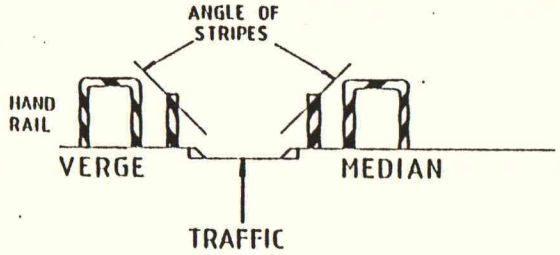
At the Fitzroy-Kent intersection, median islands are recommended for Kent St. Although there were three accidents at this location, a zebra crossing is not recommended due to the high traffic volumes and favourable geometry at the site.

For the Gladstone Rd "blackspot", regular maintenance of the shrubbery is the prescribed treatment. The pedestrian flow is not sufficient to warrant protection work.

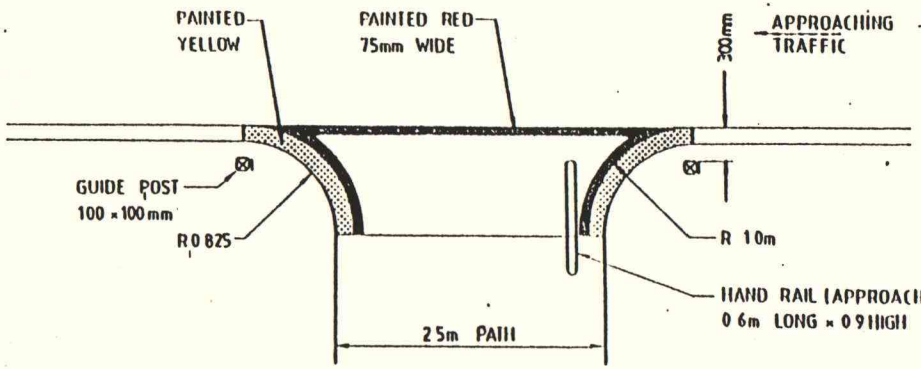
With children and the aged in the highest risk category as pedestrians the primary focus should be on improvements in areas where large concentrations of these groups occur eg. schools, homes for the aged. While schools in Rockhampton are well patrolled by "lollipop" persons now, greatly increasing the safety of our children, one site in Rockhampton I feel is need of attention. Observations taken at the Campbell-North intersection showed the elderly from the "Eventide" home had to cross this busy location without any form of assistance. The only form of control is STOP signs in North St. The majority of the people had long waits before a break in the traffic. High vehicle volumes and the width of the streets make the area a potential risk for the pedestrians.

A channelized intersection, providing refuge for the people is the ideal solution. Zebra markings should be provided on the

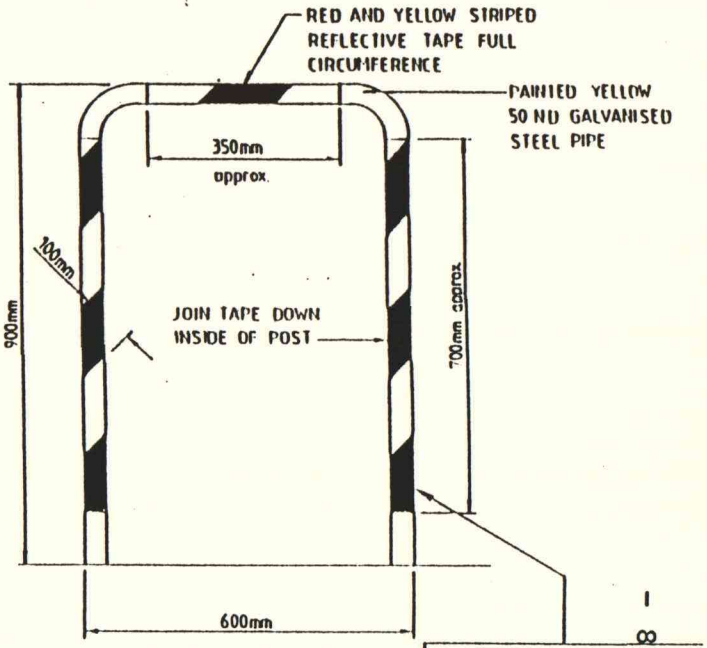




TYPICAL STRIPING VALLEY.

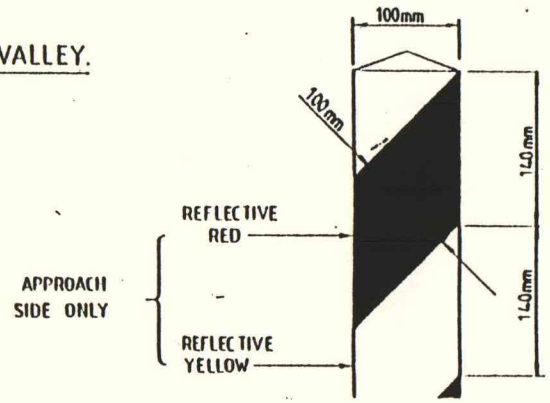


PEDESTRIAN RAMP
SCALE 1:50



HAND RAIL SCALE 1:10

ANGLED STRIPES TO CREATE 'VALLEY' FOR APPROACHING MOTORISTS



WOODEN GUIDE POSTS
SCALE 1:5
100 x 100 x 1350
SET 450 INTO GROUND CREOSOTED BASE
PAINTED YELLOW GLOSS

FIG 8.4

river leg of North St on the intersection, and also the showgrounds leg of Campbell St, with allowance for the provision of marking for the two remaining legs as well.

A system of ramps, guide posts and handrails used successfully in the City of Melville in W.A, could also be employed (see Fig 8.4)

On the whole Rockhampton is generally quite good in terms of pedestrian safety. The City Heart Mall constructed in 1984 has eliminated vehicular traffic from the busiest pedestrian location in the city. The recently constructed pathway from the UCCQ to Richardson Rd has seperated the pedestrian traffic from the busy Bruce Highway and allows a comfortable and safe walk. The school crossings are well patrolled by "lollipop" people, providing consumate safety for the children of the city.

Although it is sitting in the middle when compared to other Queensland provincial cities (Section 2) in terms of fatalities and deaths, engineering treatments are not widely needed as the majority of accidents are a result of carelessness upon the part of the pedestrian.

8.2 Education

Pedestrian safety education is necessary to reduce the risk to pedestrians on the road. Simple things like wearing light coloured clothing at night should not be overlooked when educating people in the aspects of good pedestrian behaviour. Education should be primarily targeted at schools, however public displays and demonstrations should be conducted on a planned and regular basis to continual remind people of the dangers and responsibility of using the road environment.

A public pedestrian safety campaign in Sydney during June

and July 1990 incorporated schools organising a "Pedestrian Safety Day". The schools were given an outline for the day's activities and this involved the viewing of an educational video and discussion, practical playground demonstrations, worksheet activities, a walk through around the local pedestrian environment, a design a poster competition, and a take home note for the children to mark safe routes to school. The details are given in Appendix C

This would be of great benefit here in Rockhampton as it would be anywhere. Although it results in a day lost in school it may very well prevent a life lost out of school due to poor pedestrian behaviour.

9 REFERENCES

Shinar, D
PSYCHOLOGY ON THE ROAD
John Wiley and Sons, 1978

McLean, A.J. Brewer, N.D. Sandow, B.L.
ADELAIDE IN DEPTH ACCIDENT STUDY
University of Adelaide, 1979
Part 2

Department of the Environment
PEDESTRIAN SAFETY
Her Majesty's Stationary Office, 1973

FOURTH NATIONAL SEMINAR ON PLANNING, DESIGN, AND IMPLEMENTATION OF BICYCLE AND PEDESTRIAN FACILITIES
American Society of Civil Engineers, 1976

Main Roads Department Highway Planning Branch
PREDICTION OF PEDESTRIAN TRAFFIC
Main Roads Department, 1972

Foldvary, L.A.
PEDESTRIANS, PEDALCYCLISTS, & MOTORCYCLISTS
Australian Government Publishing Service, 1973

Chapman, A.J. Foot, H.C. Wade, F.M.
PEDESTRIAN ACCIDENTS
John Wiley and Sons, 1982

Heraty, M.J.
REVIEW OF PEDESTRIAN SAFETY RESEARCH
Transport and Road Research Laboratory, UK, 1986

NAASRA
ROUNDABOUTS - A DESIGN GUIDE
NAASRA, 1986

Ribbens, H
PEDESTRIAN CASUALTIES AT ROAD INTERSECTIONS AND SUGGESTED RNGINEERINGCOUNTERMEASURES
National Institute for Transport and Road Research, 1985

NAASRA
ROADS AND PEDESTRIAN SAFETY
NAASRA, NSW, 1986

Cameron, M.H.
ACCIDENT RISKS AT ZEBRA CROSSINGS
Australian Road Research Board,1976

Homburger, W.S. Kell, J.H.
FUNDAMENTALS IN TRAFFIC ENGINEERING
Institute of Transportation Studies, USA, 1977

APPENDIX A

Attached is a listing of the computer system PEDAC. The actual program PEDAC, and units UDFS1, UDFS2, and UDFS3 are printed.

```
program pedac;
{$M 65520,0,65500}

uses
    crt,udfs1,udfs2;
var
   key:char;
   finished:boolean;

begin
     initialize;
     mainscreen;
     repeat
          cursor(off);
          key:=readkey;
          if key=#0 then
          begin
          key:=readkey;
          case key of
               f1: dohelp(1);
               f2: add_data;
               f3: view_data;
               f4: select_data;
               f5: show_acc_numbers;
               f6: options;
               f10: quit;
          end;
          end;
     until finished;
end.
```

```
unit udfs1;

interface

uses
    crt,dos;

{*** trees ***}
type
    baltype=-1..1;
    keytype=longint;
    infotype=longint;
    intlistptr=^intlist;
    strlistptr=^strlist;
    treeptr1=^tree1;
    treeptr2=^tree2;
    intlist=record
                info:longint;
                next:intlistptr;
            end;
    strlist=record
              info:string;
              next:strlistptr;
          end;
    tree1=record
              key:keytype;
              info:infotype;
              left:treeptr1;
              right:treeptr1;
          end;
    tree2=record
              key:keytype;
              next:strlistptr;
              left:treeptr2;
              right:treeptr2;
          end;

function maketree1(k:keytype;i:infotype):treeptr1;
function maketree2(k:keytype;l:strlistptr):treeptr2;
procedure leftrotation(var tree:treeptr1);
procedure rightrotation(var tree:treeptr1);
procedure insert1(var t:treeptr1;k:keytype;i:infotype;var
keyexists:boolean);
procedure insert2(var t:treeptr2;k:keytype;l:strlistptr;var
keyexists:boolean);
function search1(var t:treeptr1;k:keytype):longint;
function search2(var t:treeptr2;k:keytype):strlistptr;

{*** menus ***}

const
     spread=true;
     bunch=false;
     before=false;
     auto=99;
     enable=true;
     enabled=true;
     disable=false;
     disabled=false;
     max_id=50;
     shaddow=true;
     noshaddow=false;
```

```
      esc=#27;
      return=#13;
      f1=#59;
      f2=#60;
      f3=#61;
      f4=#62;
      f5=#63;
      f6=#64;
      f10=#68;
type
    dat=array[1..22] of string;
    bardat=dat;
    string1=string[1];
var
   LEFT,RIGHT,menuon:boolean;
   items:dat;
   num,choice,lastx,bbar,bfore,bback,bhighlight:integer;
   border,colset,fore,bar,back,highlight:array[0..max_id] of
integer;
   beenherebefore:array[0..max_id] of boolean;
   lasty:array[0..max_id] of integer;

procedure menu(id,xpos,ypos:integer;arr:dat;num:integer;var
datanum:integer;functionkeys:boolean);
procedure
helpbox(id,back,fore,line,nooflines:integer;items:dat;x,y:int
eger);
procedure editbox(a:string;var i:longint;len,l:integer;var
code:integer);
procedure editrbox(a:string;var i:real;len,dec,l:integer;var
code:integer);
procedure errorbox(a,b:string;l:integer;shad:boolean);
procedure setmenucolor(id,a,b,c,d,e:integer);
{ procedure initmenubar; }
procedure setmenubarcolor(id,f,b,bb,h:integer);
procedure menubar(id:integer;items:dat;num:integer;var
choice:integer;spread:boolean);
function firstchar(s:string):string1;
procedure windowbox(x1,y1,x2,y2:integer);

{*** utils ***}

type
    curtype=(off,big,small);
    string80=string[80];

procedure keyhalt;
procedure getesc;
procedure cursor(size:curtype);
function spaces(num:word):string80;
function nospaces(s:string):string;
function max(a,b:integer):integer;
function min(a,b:integer):integer;
function printerokay:boolean;
function strip(s:string):string;

{*** edit ***}

const
     on=1;
     mute=0;
     save=0;
```

```
     restore=1;
type
    string50=string[50];
    stringtype=string;
    date=record
              year,month,day,dayofweek:word;
         end;

procedure eddate(var d:date);
procedure screen(i:integer);
procedure print(x,y:integer;str:string);
procedure setbeep(state:integer);
procedure seteditcolor(a,b:integer);
procedure getwindow(var x1,y1,x2,y2:integer);
procedure printreal(x,y:integer;str:real;a,b:integer);
procedure printint(x,y,str:integer);
function edstr(x,y:integer;var
namestr:stringtype;max:integer):integer;
function edreal(x,y:integer;var name:real;a,b,max:integer;var
code:integer):integer;
procedure edint(x,y:integer;var name:longint;max:integer;var
code:integer);
procedure
dboxaround(filename:string;centerall:boolean;heading:string);

boxaround(filename:string;centerall:boolean;heading:string);
procedure output(var alias:text;filename:string);
procedure input(var alias:text;filename:string);
function spaceleft(s:string;width:integer):integer;
function centre(strg:string):integer;
procedure centrestr(s:string;line:integer);
function nextline:integer;
function screenwidth:integer;
function screenlength:integer;

{*** sounds ***}

const
     low=200;
     med=500;
     high=700;

procedure beep(pitch:integer);

{*** boxes ***}

const

dhlineseg='═══════════════════════════════════════════
═══════════════════════════════════════════';

hlineseg='───────────────────────────────────────────
───────────────────────────────────────────';
     vlineseg='│';
     dvlineseg='║';
     topleft='┌';
     topright='┐';
     botleft='└';
     botright='┘';
     dtopleft='╔';
     dtopright='╗';
     dbotleft='╚';
```

```
     dbotright='╝';

function horiz(len:byte):string80;
function dhoriz(len:byte):string80;
procedure hline(x1,y1,len:byte);
procedure vline(x1,y1,len:byte);
procedure dhline(x1,y1,len:byte);
procedure dvline(x1,y1,len:byte);
procedure box(x1,y1,wide,deep:byte);
procedure dbox(x1,y1,wide,deep:byte);

implementation

{*** trees ***}

procedure addlist(var ls:strlistptr;n:string);
var
   l:strlistptr;
begin
     if ls=nil then
     begin
          new(ls);
          ls^.info:=n;
          ls^.next:=nil;
     end
     else
     begin
          l:=nil;
          addlist(l,n);
     end;
end;

function maketree1(k:keytype;i:infotype):treeptr1;
var
   p:treeptr1;
begin
     new(p);
     p^.key:=k;
     p^.info:=i;
     p^.left:=nil;
     p^.right:=nil;
     maketree1:=p;
end;

function maketree2(k:keytype;l:strlistptr):treeptr2;
var
   p:treeptr2;
begin
     new(p);
     p^.key:=k;
     p^.next:=l;
     p^.left:=nil;
     p^.right:=nil;
     maketree2:=p;
end;

procedure leftrotation(var tree:treeptr1);
var
   p,q,hold:treeptr1;
begin
     q:=p^.right;
     hold:=q^.left;
```

```
     q^.left:=p;
     p^.right:=hold;
     tree:=p;
end;

procedure rightrotation(var tree:treeptr1);
var
   p,q,hold:treeptr1;
begin
     q:=p^.left;
     hold:=q^.right;
     q^.right:=p;
     p^.left:=hold;
     tree:=p;
end;

procedure insert1(var t:treeptr1;k:keytype;i:infotype;var
keyexists:boolean);
var
   x:treeptr1;
begin
     if t<>nil then
     begin
          if k=t^.key then
          begin
               t^.left:=nil;
               t^.right:=nil;
               keyexists:=true;
               exit;
          end
          else
              keyexists:=false;
          if k<t^.key then
          begin
               if t^.left=nil then
               begin
                    new(x);
                    x^.key:=k;
                    x^.info:=i;
                    t^.left:=x;
                    x^.left:=nil;
                    x^.right:=nil;
               end
               else
                   insert1(t^.left,k,i,keyexists);
          end
          else
          begin
               if t^.right=nil then
               begin
                    new(x);
                    x^.key:=k;
                    x^.info:=i;
                    t^.right:=x;
                    x^.left:=nil;
                    x^.right:=nil;
               end
               else
                   insert1(t^.right,k,i,keyexists);
          end;
     end
     else
```

```
     begin
          new(t);
          t^.key:=k;
          t^.info:=i;
          t^.left:=nil;
          t^.right:=nil;
          keyexists:=false;
     end;
end;

procedure insert2(var t:treeptr2;k:keytype;l:strlistptr;var
keyexists:boolean);
var
   x:treeptr2;
begin
     if t<>nil then
     begin
          if k=t^.key then
          begin
               t^.left:=nil;
               t^.right:=nil;
               keyexists:=true;
               exit;
          end
          else
              keyexists:=false;
          if k<t^.key then
          begin
               if t^.left=nil then
               begin
                    new(x);
                    x^.key:=k;
                    x^.next:=l;
                    t^.left:=x;
                    x^.left:=nil;
                    x^.right:=nil;
               end
               else
                   insert2(t^.left,k,l,keyexists);
          end
          else
          begin
               if t^.right=nil then
               begin
                    new(x);
                    x^.key:=k;
                    x^.next:=l;
                    t^.right:=x;
                    x^.left:=nil;
                    x^.right:=nil;
               end
               else
                   insert2(t^.right,k,l,keyexists);
          end;
     end
     else
     begin
          new(t);
          t^.key:=k;
          t^.next:=l;
          t^.left:=nil;
          t^.right:=nil;
```

```
          keyexists:=false;
     end;
end;

function search1(var t:treeptr1;k:keytype):longint;
begin
     search1:=-1;
     if t<>nil then
     begin
          if t^.key=k then
          begin
               search1:=t^.info;
          end
          else
          begin
               if k<t^.key then
               begin
                    search1:=search1(t^.left,k);
               end
               else
               begin
                    search1:=search1(t^.right,k);
               end;
          end;
     end;
end;

function search2(var t:treeptr2;k:keytype):strlistptr;
var
   x:strlistptr;
begin
     x:=nil;
     if t<>nil then
     begin
          if t^.key=k then
          begin
               search2:=t^.next;
          end
          else
          begin
               if k<t^.key then
               begin
                    search2:=search2(t^.left,k);
               end
               else
               begin
                    search2:=search2(t^.right,k);
               end;
          end;
     end;
end;

{*** menus ***}

procedure windowbox(x1,y1,x2,y2:integer);
begin
     window(x1,y1,x2,y2);
     clrscr;
     window(x1,y1,x2+2,y2);
     box(1,1,x2-x1,y2-y1-1);
     window(x1+1,y1+1,x2,y2-1);
end;
```

```
function firstchar(s:string):string1;
begin
     firstchar:=copy(s,1,1);
end;

procedure initmenubar;
begin
     lastx:=1;
end;

procedure setmenubarcolor(id,f,b,bb,h:integer);
begin
     initmenubar;
     bbar:=bb;
     DêÇ┤$,É♦L3ÿEéyø=á♦│NYàg\&
```

ÇH└ÇÿHä°ÇαL8 £ÇÇÇL †ÇÇα♦
†ÇÇÇαL°└8 └ä
†ÇÇÇL@xHÇα£†Çα

ηηη°$τ@♦τ τ αHÇÇ@-τ@τÇ | τ@êα└τL

τÇÇÇL

ηη°8xΩHHα ♦d

LH♦°î°
тССС¿кн└ç÷нä°◙ç†ç¿ѳ

ηη°<ττÿ<τ$

τÇÇÇL

ηη°Ç$tê H@tÇ$8αL

```
                    ηη°ê
@<↑┴↑┗ÇÇêHäτÇα8Çα£oolean;
begin
     finish:=false;
     x:=lastx;
     textcolor(bhighlight);
     textbackground(bbar);
     print(xpos[x],1,items[x]);
     oldx:=x;
     IF MENUON THEN
     begin
          choice:=x;
          lastx:=x;
          exit;
     end;
     repeat
     repeat
     key:=readkey;
     case key of
          #77: begin
                    x:=x+1;
                    cont:=true;
               end;
          #75: begin
                    x:=x-1;
                    cont:=true;
               end;
          #71: begin
                    x:=1;
                    cont:=true;
               end;
          #79: begin
                    x:=num;
                    cont:=true;
               end;
          #80: begin
                    finish:=true;
                    cont:=true;
               end;
          esc: begin
                    textcolor(bfore);
```

```
                    textbackground(bback);
                    print(xpos[x],1,items[x]);
                    lastx:=x;
                    choice:=27;
                    exit;
               end;
          return: begin
                    finish:=true;
                    cont:=true;
               end;
     else
     begin
          goahead:=false;
          for i:=1 to num do
          begin
               if (upcase(key)=firstchar(items[i])) or
(key=firstchar(items[i])) then
               begin
                    oldx:=x;
                    x:=i;
                    cont:=true;
                    finish:=true;
                    textcolor(bfore);
                    textbackground(bback);
                    print(xpos[oldx],1,items[oldx]);
                    textcolor(bhighlight);
                    textbackground(bbar);
                    print(xpos[x],1,items[x]);
                    oldx:=x;
               end;
          end;
     end;
     end;
     until cont;
     if x>num then x:=1;
     if x<1 then x:=num;
     if not cont or not finish then
     begin
     textcolor(bfore);
     textbackground(bback);
     print(xpos[oldx],1,items[oldx]);
     end;
     textcolor(bhighlight);
     textbackground(bbar);
     print(xpos[x],1,items[x]);
     oldx:=x;
     beenbefore:=true;
     until finish;
     choice:=x;
     lastx:=x;
end;

begin
     IF MENUON THEN
     BEGIN
          IF LEFT THEN LASTX:=LASTX-1;
          IF RIGHT THEN LASTX:=LASTX+1;
          IF LASTX<1 THEN LASTX:=NUM;
          IF LASTX>NUM THEN LASTX:=1;
     END;
     cursor(off);
     textcolor(bfore);
```

```
     textbackground(bback);
     gotoxy(1,1);
     clreol;
     xpos[1]:=3;
     largest:=0;
     sum:=0;
     for i:=1 to num-1 do
     begin
          sum:=length(items[i])+sum;
          largest:=max(largest,length(items[i]));
     end;
     if spread then
     begin
          sfactor:=80;
          step:=round(int((sfactor-xpos[1]-sum)/num-1));
     end
     else step:=2;
     for i:=2 to num do
     begin
          xpos[i]:=length(items[i-1])+xpos[i-1]+step;
     end;
     for i:=1 to num do
     begin
          print(xpos[i],1,items[i]);
     end;
     select;
     writeln;
     cursor(small);
end;

procedure
helpbox(id,back,fore,line,nooflines:integer;items:dat;x,y:int
eger);
var
   maxlen,n,x1,y1,x2,y2,h,i,j,k:integer;
   a:char;
begin
     if y=auto then y:=12;
     if x=auto then x:=40;
     getwindow(h,i,j,k);
     maxlen:=0;
     for n:=1 to nooflines do
     begin
          maxlen:=max(maxlen,length(items[n]));
     end;
     x1:=round(x-(maxlen+2)/2);
     y1:=y-round(nooflines/2);
     x2:=x1+maxlen+1;
     y2:=y1+nooflines+1;
     textbackground(0);
     window(x1+2,y1+1,x2+2,y2+1);
     clrscr;
     textbackground(back);
     window(x1,y1,x2,y2);
     clrscr;
     textcolor(line);
     window(x1,y1,x2+1,y2+1);
     dbox(1,1,maxlen+2-2,nooflines);
     textcolor(fore);
     centrestr(' Help ',1);
     for n:=1 to nooflines do
     print(2,n+1,items[n]);
```

```
          window(h,i,j,k);
          cursor(off);
          repeat
          a:=readkey;
          until a=esc;
     end;

     procedure setmenucolor(id,a,b,c,d,e:integer);
     var
        i:integer;
     begin
          menuon:=false;
          for i:=0 to 20 do beenherebefore[i]:=false;
          colset[id]:=223;
          back[id]:=b;
          fore[id]:=a;
          highlight[id]:=c;
          bar[id]:=d;
          border[id]:=e;
     end;

     procedure editbox(a:string;var i:longint;len,l:integer;var
     code:integer);
     begin

     centrestr(concat('┌',horiz(max(len+2,length(a))+1),'┐'),l-1);
entrestr(concat(a),l-1);

     centrestr(concat('│',spaces(max(length(a),len+2)+1),'│'),l);

     centrestr(concat('└',horiz(max(len+2,length(a))+1),'┘'),l+1);
otoxy(40-max(length(a),len) div 2,l);
          edint(wherex,wherey,i,len,code);
     end;

     procedure editrbox(a:string;var i:real;len,dec,l:integer;var
     code:integer);
     var
        c:integer;
     begin

     centrestr(concat('┌',horiz(max(len+2,length(a))+1),'┐'),l-1);
entrestr(concat(a),l-1);

     centrestr(concat('│',spaces(max(length(a),len+2)+1),'│'),l);

     centrestr(concat('└',horiz(max(len+2,length(a))+1),'┘'),l+1);
otoxy(40-max(length(a),len) div 2,l);
          repeat
               c:=edreal(wherex,wherey,i,len,dec,len+dec,code);
          until c=0;
     end;

     procedure errorbox(a,b:string;l:integer;shad:boolean);
     function padifsmall(b:string):string;
     var
        x:integer;
     begin
          x:=(length(a)-length(b))div 2;
          b:=concat(spaces(x),b,spaces(length(a)-length(b)-x));
          padifsmall:=b;
     end;
```

```
begin

centrestr(concat('┌',dhoriz(max(length(b),length(a))+2),'┐'),
l-1);
     centrestr(concat('│ ',padifsmall(b),' │'),l);
     centrestr(concat(a),l-1);

centrestr(concat('└',dhoriz(max(length(b),length(a))+2),'┘'),
l+1);
end;

procedure menu(id,xpos,ypos:integer;arr:dat;num:integer;var
datanum:integer;functionkeys:boolean);
var
   maxlen,i,x,y,x1,y1,wide,deep,max,oldx,oldy:integer;
   op:array[1..3] of string;
   a1,b1,a2,b2:integer;
   chosen:boolean;
   key:char;


procedure select;
var
   inkey:char;
   i:integer;
begin
     if colset[id]<>223 then
     begin
          i:=id;
               highlight[i]:=15;
               fore[i]:=15;
               back[i]:=7;
               bar[i]:=0;
     end;
     chosen:=false;
     if not beenherebefore[id] then lasty[id]:=1;
     y:=lasty[id];
     beenherebefore[id]:=true;
     x:=1;
     if y<1 then y:=num;
     if y>num then y:=1;
     window(x1,y1+1,x1+wide-2,y1+deep);
     textcolor(fore[id]);
     textbackground(back[id]);
     for i:=1 to num do print(x,i,arr[i]);
     textcolor(highlight[id]);
     textbackground(bar[id]);
     print(x,y,arr[y]);
     repeat
          key:=readkey;
          case key of
               #0: begin
                        key:=readkey;
                        case key of


               #73: begin
                         oldy:=y;
                         y:=1;
                    end;
               #81: begin
                         oldy:=y;
```

```
                    y:=num;
               end;
          #72: begin
                    oldy:=y;
                    y:=y-1;
               end;
          f1: begin
                    chosen:=true;
                    datanum:=59;
               end;
          #75: BEGIN
                    CHOSEN:=TRUE;
                    DATANUM:=0;
                    LEFT:=TRUE;
                    RIGHT:=FALSE;
                    menuon:=true;
                    lasty[id]:=y;
               END;
          #77: BEGIN
                    CHOSEN:=TRUE;
                    DATANUM:=0;
                    LEFT:=FALSE;
                    RIGHT:=TRUE;
                    menuon:=true;
                    lasty[id]:=y;
               END;
          #80: begin
                    oldy:=y;
                    y:=y+1;
               end;
               end;
               end;
               return: begin
                    chosen:=true;
                    datanum:=y;
                    lasty[id]:=y;
                    menuon:=false;
               end;
               esc: begin
                    chosen:=true;
                    datanum:=0;
                    lasty[id]:=y;
                    menuon:=false;
                    choice:=27;
               end;
     else
     begin
          for i:=1 to num do
          begin
               if (upcase(key)=firstchar(arr[i])) or
(key=firstchar(arr[i])) then
               begin
                    chosen:=true;
                    datanum:=i;
                    lasty[id]:=i;
               end;
          end;
          if functionkeys then
          begin
          case key of
               f1: begin
                         chosen:=true;
```

```
                    datanum:=59;
                end;
            f2: begin
                    chosen:=true;
                    datanum:=60;
                end;
            f3: begin
                    chosen:=true;
                    datanum:=61;
                end;
            f4: begin
                    chosen:=true;
                    datanum:=62;
                end;
            f5: begin
                    chosen:=true;
                    datanum:=63;
                end;
            f6: begin
                    chosen:=true;
                    datanum:=54;
                end;
            #65: begin
                    chosen:=true;
                    datanum:=65;
                end;
            #66: begin
                    chosen:=true;
                    datanum:=66;
                end;
            #67: begin
                    chosen:=true;
                    datanum:=67;
                end;
            f10: begin
                    chosen:=true;
                    datanum:=68;
                end;
        end;
    end;

    end;
    end;
    if not chosen then
    begin
        if y<1 then y:=num;
        if y>num then y:=1;
        window(x1,y1+1,x1+wide-2,y1+deep);
        textcolor(fore[id]);
        textbackground(back[id]);
        for i:=1 to num do print(x,i,arr[i]);
{        print(x,pred(y),arr[pred(y)]);}
        textcolor(highlight[id]);
        textbackground(bar[id]);
        print(x,y,arr[y]);
        lasty[id]:=y;
        beenherebefore[id]:=true;
        gotoxy(x,y);
    end;
until chosen;
end;
```

```
begin
     IF MENUON THEN
     BEGIN
          CHOSEN:=TRUE;
          DATANUM:=Y;
     END;
     getwindow(a1,b1,a2,b2);
     cursor(off);
     maxlen:=0;
     for i:=1 to num do
     begin
          if length(arr[i])>maxlen then
maxlen:=length(arr[i]);
     end;
     if ypos=99 then y1:=round(int(0.5*(25-num)))-1
     else y1:=ypos;
     if xpos=99 then x1:=round(0.5*(80-maxlen))-1
     else x1:=xpos;
     wide:=maxlen+1;
     deep:=num+1;
     textbackground(0);
     window(x1+1,y1+1,x1+wide+1,y1+deep+1);
     clrscr;
     window(x1-1,y1,x1+wide-1,y1+deep);
     textcolor(fore[id]);
     textbackground(back[id]);
     clrscr;
     window(x1,y1+1,x1+wide-2,y1+deep);
     textcolor(border[id]);
     textbackground(back[id]);
     window(1,1,80,25);
     dbox(x1-1,y1,wide-1,deep-1);
     textcolor(fore[id]);
     oldx:=1;
     window(x1,y1+1,x1+wide-2,y1+deep);
     select;
     window(a1,b1,a2,b2);
     cursor(small);
     normvideo;
end;

{*** utils ***}

function printerokay:boolean;
begin

end;



procedure getesc;
var
   key:char;
begin
     repeat
          key:=readkey;
     until key=esc;
end;

function max(a,b:integer):integer;
begin
     if a<b then max:=b else max:=a;
```

```
end;

function min(a,b:integer):integer;
begin
     if a<b then min:=a else min:=b;
end;

procedure keyhalt;
var
   trash:char;
begin
     repeat until keypressed;
     trash:=readkey;
     if trash=#0 then trash:=readkey;
end;

procedure cursor(size:curtype);
var
   regs:registers;
begin
     with regs do
     begin
          AX:=$100;
          case size of
               off: CX:=$3030;
               small: CX:=$607;
               big: CX:=$0F;
          end;
          intr($10,regs);
     end;
end;

function spaces(num:word):string80;
const
     space='
                                        ';
begin
     spaces:=copy(space,1,num);
end;

function nospaces(s:string):string;
var
   ch:string;
   i:integer;
begin
     i:=1;
     ch:=' ';
     while ch=' ' do
     begin
          ch:=s[i];
          s:=copy(s,i,length(s));
          inc(i);
     end;
     nospaces:=s;
end;

function strip(s:string):string;
var
   l,i:integer;
   ch:char;
begin
     l:=length(s);
```

```
     ch:=' ';
     i:=1;
     while ch=' ' do
     begin
          ch:=s[i];
          s:=copy(s,i,l);
          inc(i);
     end;
     ch:=' ';
     l:=length(s);
     i:=l;
     while ch=' ' do
     begin
          ch:=s[i];
          s:=copy(s,1,i);
          dec(i);
     end;
     strip:=s;
end;

{*** edit ***}

var
   beepon:boolean;
   edcol,edbcol:integer;

procedure eddate(var d:date);
begin

end;

procedure output(var alias:text;filename:string);
begin
     {$I-}
     assign(alias,filename);
     rewrite(alias);
     {$I+}
end;

procedure input(var alias:text;filename:string);
begin
     {$I-}
     assign(alias,filename);
     reset(alias);
     {$I+}
end;

function spaceleft(s:string;width:integer):integer;
begin
     spaceleft:=width-length(s);
end;

procedure print(x,y:integer;str:string);
begin
     gotoxy(x,y);
     write(str);
end;

procedure setbeep(state:integer);
begin
     if state=1 then beepon:=true;
     if state=0 then beepon:=false;
```

```
end;

procedure printreal(x,y:integer;str:real;a,b:integer);
begin
     gotoxy(x,y);
     write(str:a:b);
end;

procedure printint(x,y,str:integer);
begin
     gotoxy(x,y);
     write(str);
end;

procedure seteditcolor(a,b:integer);
begin
     edcol:=a;
     edbcol:=b;
end;

function edstr(x,y:integer;var
namestr:stringtype;max:integer):integer;
var
   key:char;
   oldstr:string;
   firsttime,skip:boolean;
const
     spaces='
          ';
begin
     skip:=false;
     oldstr:=namestr;
     textcolor(edcol);
     textbackground(edbcol);

print(x,y,concat(namestr,copy(spaces,1,max-length(namestr))))
;
     namestr:='';
     gotoxy(x,y);
     firsttime:=true;
    repeat
     key:=readkey;
     if (key=return) and (firsttime) then
     begin
          edstr:=0;
          namestr:=oldstr;
          exit;
     end;
     firsttime:=false;
     case key of
          chr(8): begin
                       gotoxy(wherex-1,y);

namestr:=copy(namestr,1,length(namestr)-1);
                       print(x,y,copy(spaces,1,max));
                       print(x,y,namestr);
                       gotoxy(length(namestr)+x,y);
                  end;
          #0: begin
                   key:=readkey;
                   case key of
                        f1: begin
```

```
                                edstr:=59;
                                exit;
                           end;
                      f2: begin
                                edstr:=60;
                                exit;
                           end;
                      f3: begin
                                edstr:=61;
                                exit;
                           end;
                      f4: begin
                                edstr:=62;
                                exit;
                           end;
                      f5: begin
                                edstr:=63;
                                exit;
                           end;
                      f6: begin
                                edstr:=64;
                                exit;
                           end;
                      #65: begin
                                edstr:=65;
                                exit;
                           end;
                      #66: begin
                                edstr:=66;
                                exit;
                           end;
                      #67: begin
                                edstr:=67;
                                exit;
                           end;
                      f10: begin
                                edstr:=68;
                                exit;
                           end;
                      #75,#83,chr(8): begin
                                         gotoxy(wherex-1,y);

namestr:=copy(namestr,1,length(namestr)-1);

print(x,y,copy(spaces,1,max));
                                         print(x,y,namestr);

gotoxy(length(namestr)+x,y);
                                         skip:=true;
                                    end
                          end;
                end;

     end;
                    case key of
          '
',',','.','a'..'z','A'..'Z','0'..'9',',',',','?','/','=','!','@','#',
'*' :begin
                                                    if not
skip then
                                                    begin
```

```
key:=upcase(key);
                                                          if
length(namestr)<max then namestr:=concat(namestr,key);
                                                          if
length(namestr)>max then
                                                          begin

 beep(high);}

exit;
                                                          end;

print(x,y,copy(spaces,1,max));

print(x,y,namestr);

gotoxy(x+length(namestr),y);
                                                    end;

skip:=false;
                                                  end;

          else
              begin
                   case key of
                        return: begin
{                                beep(high);}
                                edstr:=0;
                                exit;
                           end;
          esc: begin
                    namestr:=oldstr;
                    exit;
               end;
              end;
     end;
     end;
    until false;
end;

procedure edrealstr(x,y:integer;var
namestr:string;max:integer;var code:integer);
var
   key:char;
const
     spaces='
          ';
begin
     gotoxy(x,y);
     namestr:='';
    repeat
     code:=0;
     key:=readkey;
     case key of
          return: begin
{                  beep(high);}
                    exit;
               end;
          f1: begin
                    code:=59;
                    exit;
               end;
```

```
          f2: begin
                    code:=60;
                    exit;
               end;
          esc: begin
                    code:=27;
                    exit;
               end;
          #75,#83,chr(8): begin
                         gotoxy(wherex-1,y);

namestr:=copy(namestr,1,length(namestr)-1);
                         print(x,y,copy(spaces,1,max));
                         print(x,y,namestr);
                         gotoxy(length(namestr)+x,y);
                    end
          else
              begin
                   case key of
                        '0'..'9','.': begin
                                           if length(namestr)<max
then namestr:=concat(namestr,key);

print(x,y,copy(spaces,1,max));
                                           print(x,y,namestr);

gotoxy(x+length(namestr),y);
                                           if
length(namestr)>=max then exit;
                                      end;
                    end;
               end;
     end;
    until false;
end;

function edreal(x,y:integer;var name:real;a,b,max:integer;var
code:integer):integer;

var
   strg,ch,s,rev:string;
   addon,old:real;
   c,j,key:integer;
begin
     textcolor(edcol);
     textbackground(edbcol);
     old:=name;
     addon:=0;
     name:=0;
     strg:='';
     edrealstr(x,y,rev,max,code);
     val(rev,name,c);
{     case key of
         59: begin
                   edreal:=59;
              end;
     end;}

     edreal:=key;
end;

procedure edint(x,y:integer;var name:longint;max:integer;var
```

```
code:integer);
var
   num:real;
   c:integer;
begin
     cursor(small);
     textcolor(edcol);
     textbackground(edbcol);
     c:=edreal(x,y,num,max,0,max,code);
     if code=27 then exit;
     if num=0 then
     begin
          name:=0;
          exit;
     end;
     name:=round(num);
     cursor(off);
end;

procedure
dboxaround(filename:string;centerall:boolean;heading:string);

   firstline,xpos,ypos,i,n,max:integer;
   line:array[1..22] of string;
   f:text;
begin
     {$I-}
          assign(f,filename);
          reset(f);
          if ioresult<>0 then exit;
     {$I+}
     i:=1;
     max:=0;
     while not eof(f) do
     begin
          readln(f,line[i]);
          if length(line[i])>max then max:=length(line[i]);
          if max>78 then line[i]:=copy(line[i],1,78);
          i:=i+1;
     end;
     n:=i;
     if n>21 then n:=21;
     firstline:=round(int((25-n)/2));
     clrscr;
     if centerall then
     begin
          for i:=1 to n do
          begin
               xpos:=round(40-length(line[i])/2);
               ypos:=firstline+i;
               print(xpos,ypos,line[i]);
          end;
     end
     else
     begin
          xpos:=round(40-max/2);
          for i:=1 to n do
          begin
               ypos:=firstline+i;
               print(xpos,ypos,line[i]);
               writeln;
          end;
```

```
     end;
     xpos:=round(40-max/2);
     dbox(xpos-2,firstline-1,max+2,n);
     print(round(40-length(heading)/2),firstline-1,heading);
     close(f);
end;

procedure
boxaround(filename:string;centerall:boolean;heading:string);
var
   firstline,xpos,ypos,i,n,max:integer;
   line:array[1..22] of string;
   f:text;
begin
     {$I-}
          assign(f,filename);
          reset(f);
          if ioresult<>0 then exit;
     {$I+}
     i:=1;
     max:=0;
     while not eof(f) do
     begin
          readln(f,line[i]);
          if length(line[i])>max then max:=length(line[i]);
          if max>78 then line[i]:=copy(line[i],1,78);
          i:=i+1;
     end;
     n:=i;
     if n>21 then n:=21;
     firstline:=round(int((25-n)/2));
     clrscr;
     if centerall then
     begin
          for i:=1 to n do
          begin
               xpos:=round(40-length(line[i])/2);
               ypos:=firstline+i;
               print(xpos,ypos,line[i]);
          end;
     end
     else
     begin
          xpos:=round(40-max/2);
          for i:=1 to n do
          begin
               ypos:=firstline+i;
               print(xpos,ypos,line[i]);
               writeln;
          end;
     end;
     xpos:=round(40-max/2);
     box(xpos-2,firstline-1,max+2,n);
     print(round(40-length(heading)/2),firstline-1,heading);
     close(f);
end;

procedure getwindow(var x1,y1,x2,y2:integer);

begin
     x1:=lo(windmin)+1;
     y1:=hi(windmin)+1;
```

```
     x2:=lo(windmax)+1;
     y2:=hi(windmax)+1;
end;

function screenwidth:integer;
var
   x1,x2,y1,y2:integer;
begin
     getwindow(x1,y1,x2,y2);
     screenwidth:=round(x2-x1);
end;

function screenlength:integer;
var
   x1,x2,y1,y2:integer;
begin
     getwindow(x1,y1,x2,y2);
     screenlength:=round(y2-y1);
end;

function centre(strg:string):integer;
begin
     centre:=round(screenwidth-length(strg)/2);
end;

procedure centrestr(s:string;line:integer);
var
   xpos,len,width:integer;
begin
     len:=length(s);
     xpos:=round(screenwidth/2-len/2);
     print(xpos,line,s);
end;

function nextline:integer;
var
   lastline,oldline,oldx:integer;
begin
     oldline:=wherey;
     oldx:=wherex;
     lastline:=screenlength;
     if oldline>=lastline then
     begin
          gotoxy(oldx,oldline);
          writeln;
          nextline:=wherey;
          exit;
     end;
     if oldline<lastline then
     begin
          nextline:=oldline+1;
     end;
end;

procedure screen(i:integer);

const
     bseg=$0040;
     vbiosofs=$49;
type
    videorecs=record
                    videomode:byte;
```

```
                    numcol,screensize,memoryofs:word;
                    cursorarea:array[0..7] of word;
                    cursormode:word;
                    currentpage:byte;
                    videoboardaddr:word;
                    currentmode,currentcolor:byte;
        end;
    var
       buffsize,initcmode:word;
       npxflg:boolean;
       buffer:array[0..8191] of word;
       npxstate:array[0..93] of byte;
       initvideo:byte;
       regs:registers;
       videorec:videorecs;
       keylock:byte;
       scrnseg,numchr:word;
       a,b,c,d:integer;
    begin
         getwindow(a,b,c,d);
         if i=0 then
         begin
              swapvectors;

    move(ptr(bseg,vbiosofs)^,videorec,sizeof(videorec));
              with videorec,regs do
              begin
                   if (videomode>7) or (screensize>buffsize) then
begin
                        swapvectors;
                        exit;
                   end;
                   keylock:=mem[bseg:$0017];
                   if videomode=7 then scrnseg:=$B000
                   else scrnseg:=$B800;

    move(ptr(scrnseg,memoryofs)^,buffer,screensize);
                   AX:=initvideo;
                   if (videomode>4) and (videomode<=6) then
    intr($10,regs);
                   AX:=$0800; { was $0500 }
                   intr($10,regs);
                   CX:=initcmode;
                   AH:=2; { was 1 }
                   intr($10,regs);
                   if npxflg then inline($98/ $DD/$36/>npxstate);
end;
              end;
         if i=1 then
         begin
              window(a,b,c,d);
              if npxflg then inline($98/$DD/$36/>npxstate);
              mem[bseg:$17]:=(mem[bseg:$17] and $0F) or (keylock
    and $F0);
            with videorec,regs do
              begin
              if mem[bseg:vbiosofs] <> videomode then

              begin
                   AX:=videomode;
                   intr($10,regs);
              end;
```

```
            AH:=1;
            CX:=cursormode;
            intr($10,regs);
            AH:=5;
            AL:=currentpage;
            intr($10,regs);
            AH:=2;
            BH:=currentpage;
            DX:=cursorarea[currentpage];
            intr($10,regs);
            move(buffer,ptr(scrnseg,memoryofs)^,screensize);
          end;
            swapvectors;
            exit;
      end;

      with regs do
      begin
            intr($11,regs);
            npxflg:=(AL and 2)=2;
            AH:=15;
            intr($10,regs);
            initvideo:=AL;
            AH:=3;
            BH:=0;
            intr($10,regs);
            initcmode:=CX;
      end;

      buffsize:=sizeof(buffer);
end;

{*** sounds ***}

procedure beep(pitch:integer);
begin
      sound(pitch);
      delay(100);
      nosound;
end;

{*** boxes ***}

function horiz(len:byte):string80;
begin
      horiz:=copy(hlineseg,1,len);
end;

function dhoriz(len:byte):string80;
begin
      dhoriz:=copy(dhlineseg,1,len);
end;

procedure hline(x1,y1,len:byte);
begin
      gotoxy(x1,y1);
      write(horiz(len));
end;

procedure vline(x1,y1,len:byte);
var
   i:byte;
```

```
begin
     for i:=y1 to y1+len do
     begin
          gotoxy(x1,i);
          write(vlineseg);
     end;
end;

procedure dvline(x1,y1,len:byte);
var
   i:byte;
begin
     for i:=y1 to y1+len do
     begin
          gotoxy(x1,i);
          write(dvlineseg);
     end;
end;

procedure dhline(x1,y1,len:byte);
begin
     gotoxy(x1,y1);
     write(dhoriz(len));
end;

procedure box(x1,y1,wide,deep:byte);
begin
     gotoxy(x1,y1);
     write(topleft+horiz(wide)+topright);
     vline(x1,y1+1,deep);
     vline(x1+1+wide,y1+1,deep);
     gotoxy(x1,y1+deep+1);
     write(botleft+horiz(wide)+botright);
end;

procedure dbox(x1,y1,wide,deep:byte);
begin
     gotoxy(x1,y1);
     write(dtopleft+dhoriz(wide)+dtopright);
     dvline(x1,y1+1,deep);
     dvline(x1+1+wide,y1+1,deep);
     gotoxy(x1,y1+deep+1);
     write(dbotleft+dhoriz(wide)+dbotright);
end;

end.
```

```
unit udfs2;

interface

uses
    crt,dos,udfs1,udfs3;

procedure readtrees;
procedure readcolors;
procedure mainscreen;
procedure dohelp(id:integer);
procedure add_data;
procedure view_data;
procedure select_data;
procedure show_acc_numbers;
procedure options;
procedure quit;
procedure initialize;

implementation

procedure dohelp(id:integer);
begin
     udfs3.dohelp(id);
end;

procedure readcolors;
var
   f:text;
   code,color:integer;
   line:string;
begin
     assign(f,colorfile);
     {$I-}
     reset(f);
     {$I+}
     if ioresult<>0 then
     begin
          textbackground(0);
          clrscr;
          color:=1;
          changecolors(color);
          yn:=false;
          error(' Error ',concat('File ',colorfile,' not
found. Press ESC. '),12,yn);
     end
     else
     begin
          readln(f,line);
          val(line,color,code);
          if code<>0 then
          begin
               color:=1;
               changecolors(color);
               yn:=false;
               error(' Error ',' Error in Addpedac.col file.
Contact programmer. Press ESC. ',12,yn);
          end
          else
              changecolors(color);
     end;
```

```
        end;


        procedure readtrees;
        var
           n,number:integer;
           line:string;
           add:boolean;
           treefile:text;
           l:strlistptr;

        begin
             {$I-}
             reset(datafile);
             {$I+}
             if ioresult<>0 then
             begin
                  datafilenotfound:=true;
             end
             else
                 datafilenotfound:=false;
             n:=0;
             if not datafilenotfound then
             begin
             while not eof(datafile) do
             begin
                  read(datafile,data);
                  insert1(maintree,data.numeric[acc_no],n,keyexists);
inc(n);
             end;
             close(datafile);
             end;

             {$I-}
             assign(treefile,menusfile);
             reset(treefile);
             {$I+}
             if ioresult<>0 then
             begin
                  yn:=false;
                  error(' Error ',concat('File ',menusfile,' not
        found. Contact programmer. Press ESC'),12,yn);
                  clrscr;
                  halt(1);
             end;
             n:=0;
             line:='';
             while not eof(treefile) do
             begin
                  if (line='***') or (n=0) then
                     add:=true
                  else
                      add:=false;
                  readln(treefile,line);
                  if add then
                  begin
                       l:=nil;
                       val(line,number,code);
                       readln(treefile,line);
                       while line<>'***' do
                       begin
                            addstrlist(l,line);
```

```
                    readln(treefile,line);
               end;
               insert2(menustree,number,l,keyexists);
          end;
          inc(n);
     end;
     close(treefile);

     {$I-}
     assign(treefile,helpfile);
     reset(treefile);
     {$I+}
     helpfilenotfound:=false;
     if ioresult<>0 then
     begin
          helpfilenotfound:=true;
          exit;
     end;
     n:=0;
     while not eof(treefile) do
     begin
          if (line='***') or (n=0) then
             add:=true
          else
              add:=false;
          readln(treefile,line);
          if add then
          begin
               l:=nil;
               val(line,number,code);

               readln(treefile,line);
               while line<>'***' do
               begin
                    addstrlist(l,line);
                    readln(treefile,line);
               end;
               insert2(helptree,number,l,keyexists);
          end;
          inc(n);
     end;
     close(treefile);
end;

procedure mainscreen;
begin
     window(1,1,80,25);
     textbackground(0);
     clrscr;
     textcolor(mainfore);
     textbackground(mainback);
     window(5,4,75,22);
     clrscr;
     window(1,1,80,25);
     dbox(5,4,70,18);
     centrestr(' P E D E S T R I A N   A C C I D E N T   S Y
S T E M ',4);
     centrestr('
├──────────────────────────────────────────────────────────
──────────┤',7);
     textbackground(0);
     gotoxy(3,7);
```

```
                write('  ');
                textbackground(mainback);
                centrestr(title,6);
                centrestr('[F2]  : Add Data for an Accident       ',10);
ntrestr('[F3]  : View Accident Data
           ',12);
                centrestr('[F4]  : Select Accident Data          ',14);
ntrestr('[F5]  : Show List of Accident Numbers
           ',16);
                centrestr('[F6]  : Options                       ',18);
ntrestr('[F10] : Exit Pedestrian Accident
           System',20);
                finished:=false;
           end;

           procedure add_data;
           var
              recno,n:integer;
           begin
                counter:=starty;
                yn:=false;
                repeat
                     finished:=false;
                     yn:=false;
                     repeat
                          clrscr;
                          mainscreen;
                          writescreen(hidedata,pg1);
                          n:=1;
                          get(n,code);
                          if code=-1 then
                          begin
                               mainscreen;
                               exit;
                          end;

           recno:=search1(maintree,data.numeric[acc_no]);
                          if recno=-1 then
                             finished:=true
                          else
                          begin
                               finished:=false;
                               error(' Error ',' Accident Number
           already exist. Press ESC. ',12,yn);
                          end;
                          inc(n);
                     until finished;
                     while n<=(no_of_fields div 2) do
                     begin
                          get(n,code);
                          inc(n);
                     end;
                     writescreen(hidedata,pg2);
                     while n<=no_of_fields do
                     begin
                          get(n,code);
                          inc(n);
                     end;
                     yn:=true;
                     error('Verify','Is the above information correct ?
           (Y/N)',12,yn);
                     cursor(small);
```

```
     until yn;
     if datafilenotfound then
        rewrite(datafile)
     else
     begin
          reset(datafile);
          seek(datafile,filesize(datafile));
     end;

insert1(maintree,data.numeric[acc_no],filepos(datafile),keyex
ists);
     if not keyexists then
        writerecord(data);
     close(datafile);
     mainscreen;
end;

procedure view_data;
var
   pos,recno:integer;
   no:longint;
begin
     begin
          edintbox('Enter accident
number',no,acc_no_length,boxline-1);
          {$I-}
          reset(datafile);
          {$I+}
          if ioresult<>0 then
          begin
               writeln('data file not found.');
               halt;
          end;
          recno:=search1(maintree,no);
          if recno=-1 then
          begin
               yn:=false;
               error(' Error ',' Accident number does not
exist. Press ESC. ',12,yn);
               exit;
          end;
          seek(datafile,recno);
          read(datafile,data);
          n:=0;
          writescreen(showdata,pg1);
          repeat
               key:=readkey;
               case key of
                    pgup: writescreen(showdata,pg1);
                    pgdn: writescreen(showdata,pg2);
                    f1: dohelp(1);
               end;
          until key=esc;
     end;
     mainscreen;
end;

procedure options;
begin
     getlist(menusfile,26);
     menu(26,55,12,list,n,choice,false);
     case choice of
```

```
          1: setcolors;
          2: shell;
     end;
     mainscreen;
end;

procedure select_data;

procedure exiting;
begin

end;

var
   menuchoice,field,fieldchoice,no,action:integer;
   l,node:intlistptr;
   js:stringtype;
   s:string;
   j:longint;
   jr:real;

const
     d=5;

begin
     {$I-}
     reset(datafile);
     {$I+}
     yn:=false;
     if ioresult<>0 then
     begin
          error(' Error ',' Data file not found. Press ESC.
',12,yn);
          exit;
     end;
     recordcount:=0;
     linecount:=0;
     count:=0;
     l:=nil;

     getlist(menusfile,28);
     menu(28,5,7,list,n,choice,false);
     case choice of
          0: begin
                  mainscreen;
                  exit;
             end;

          1: begin
                  no:=29;
                  getlist(menusfile,no);
                  menu(no,9,4,list,n,choice,false);
                  menuchoice:=choice;
                  case choice of
                       0: begin
                               mainscreen;
                               exit;
                          end;
                       1: field:=3;
                       2: field:=5;
                       3: field:=6;
                       4: field:=7;
```

```
                    5: field:=8;
                    6: field:=9;
                    7: field:=10;
                    8: field:=11;
                    9: field:=12;
                    10: field:=13;
                    11: field:=15;
                    12: field:=17;
                    13: field:=18;
                    14: field:=19;
                    15: field:=20;
                    16: field:=22;
                    17: field:=23;
                    18: field:=24;
                    19: field:=25;
                end;
                getlist(menusfile,field);
                menu(field,19,6,list,n,fieldchoice,false);
                if fieldchoice=0 then
                begin
                    mainscreen;
                    exit;
                end;
                while not eof(datafile) do
                begin
                    read(datafile,data);
                    if data.numeric[field]=fieldchoice
then
                    begin

addintlist(l,data.numeric[acc_no]);
                        inc(recordcount);
                    end;
                end;
                textcolor(modfore);
                textbackground(modback);
                windowbox(x1,y1,x2,y2);
                gotoxy(1,1);
                writelist(l);
                windowbox(5,19,75,25);
                gotoxy(1,1);
                str(recordcount,s);
                centrestr(concat('The Total number of
records found with'),1);

centrestr(concat('"',getchoice(menusfile,29,menuchoice),'"'),
2);
                centrestr('matching ',3);

centrestr(concat('"',getchoice(menusfile,field,fieldchoice),'
"'),4);
                centrestr(concat('is ',s),5);
                window(1,1,80,25);
                centrestr(' Press ESC for main menu. ',25);
getesc;
                textcolor(mainfore);
                textbackground(mainback);
            end;
          2: begin
                no:=30;
                getlist(menusfile,no);
                menu(no,9,6,list,n,choice,false);
```

```
                menuchoice:=choice;
                case choice of
                     0: begin
                             mainscreen;
                             exit;
                        end;
                     1: field:=1;
                     2: field:=2;
                     3: field:=4;
                     4: field:=14;
                     5: field:=16;
                     6: field:=21;
                     7: field:=26;
                     8: field:=27;
                     9: field:=28;
                end;
                getlist(menusfile,31);
                if (choice<>8) and (choice<>9) then
                begin
menu(31,15,choice+7,list,n,action,false);
                     if action=0 then
                     begin
                          mainscreen;
                          exit;
                     end;
                end
                else
                    action:=1;
                case choice of
                     1: begin
                             id:=acc_no;
                             edintbox('Enter accident
number',j,acc_no_length,17);
                             while not eof(datafile) do
                             begin
                                  read(datafile,data);
                                  case action of
                                       0: begin
                                               mainscreen;
                              a•'Å|►◄►Çñη
?◘Ç┤┬çåPîηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηηη°₡
```

puuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuu

ππππππππππππππππππππππππππππππππππππππππππππππππ°

î°ÿαÇE ᴸ

ÿ♦

1. ..... .

```
ππππππππππππππππππππππππππππππππππππππππππππππππππππππππππππ°αÿÿ
↑↑ ∂~8~↑ÿa.numeric[acc_no]);

 inc(recordcount);
                                                        end;
                                                    end;
                                                2: begin
                                                        if
data.numeric[id]<=j then
                                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                        end;
```

```
                                        end;
                                    3: begin
                                            if
data.numeric[id]<j then
                                            begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                            end;
                                        end;
                                    4: begin
                                            if
data.numeric[id]>j then
                                            begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                            end;
                                        end;
                                    5: begin
                                            if
data.numeric[id]>=j then
                                            begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                            end;
                                        end;
                                    6: begin
                                            if
data.numeric[id]<>j then
                                            begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                            end;
                                        end;
                                end;
                            end;
                        end;
                    2: begin
                            id:=date_of_acc;
                            textcolor(errfore);
                            textbackground(errback);
                            errorbox(' Enter date of
accident','',16,shaddow);
                            gotoxy(29,16);
                            readdate(j,code);
                            while not eof(datafile) do
                            begin
                                read(datafile,data);
                                case action of
                                    1: begin
                                            if
data.numeric[id]=j then
                                            begin

addintlist(l,data.numeric[acc_no]);
```

```
inc(recordcount);
                                                end;
                                            end;
                                        2: begin
                                                if
data.numeric[id]<=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        3: begin
                                                if
data.numeric[id]<j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        4: begin
                                                if
data.numeric[id]>j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        5: begin
                                                if
data.numeric[id]>=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        6: begin
                                                if
data.numeric[id]<>j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                    end;
                                end;
                            end;
                        3: begin
                                id:=hour_of_acc;
                                edintbox('Enter hour of
accident',j,2,17);
                                while not eof(datafile) do
```

```
                    begin
                        read(datafile,data);
                        case action of
                            1: begin
                                    if
data.numeric[id]=j then
                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                    end;
                               end;
                            2: begin
                                    if
data.numeric[id]<=j then
                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                    end;
                               end;
                            3: begin
                                    if
data.numeric[id]<j then
                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                    end;
                               end;
                            4: begin
                                    if
data.numeric[id]>j then
                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                    end;
                               end;
                            5: begin
                                    if
data.numeric[id]>=j then
                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                    end;
                               end;
                            6: begin
                                    if
data.numeric[id]<>j then
                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                    end;
```

```
                                        end;
                                    end;
                                end;
                            end;
                        4: begin
                                id:=speed_limit;
                                edintbox('Enter speed
limit',j,3,17);
                                while not eof(datafile) do
                                begin
                                    read(datafile,data);
                                    case action of
                                        1: begin
                                                if
data.numeric[id]=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        2: begin
                                                if
data.numeric[id]<=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        3: begin
                                                if
data.numeric[id]<j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        4: begin
                                                if
data.numeric[id]>j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        5: begin
                                                if
data.numeric[id]>=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
```

```
                                        6: begin
                                                if
data.numeric[id]<>j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                    end;
                                end;
                            end;
                        5: begin
                                id:=ped_age;
                                edintbox('Enter pedestrian
age',j,3,17);
                                while not eof(datafile) do
                                begin
                                     read(datafile,data);
                                     case action of
                                          1: begin
                                                if
data.numeric[id]=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                          2: begin
                                                if
data.numeric[id]<=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                          3: begin
                                                if
data.numeric[id]<j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                          4: begin
                                                if
data.numeric[id]>j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                          5: begin
```

```
                                        if
data.numeric[id]>=j then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                6: begin
                                        if
data.numeric[id]<>j then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                end;
                            end;
                        6: begin
                                id:=driver_age;
                                edintbox('Enter driver
age',j,3,17);
                                while not eof(datafile) do
                                begin
                                    read(datafile,data);
                                    case action of
                                        1: begin
                                                if
data.numeric[id]=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        2: begin
                                                if
data.numeric[id]<=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        3: begin
                                                if
data.numeric[id]<j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        4: begin
                                                if
```

```
data.numeric[id]>j then
                                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                    end;
                                               end;
                                          5: begin
                                                    if
data.numeric[id]>=j then
                                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                    end;
                                               end;
                                          6: begin
                                                    if
data.numeric[id]<>j then
                                                    begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                    end;
                                               end;
                                     end;
                                end;
                           end;
                      7: begin
                           id:=driver_bac;
                           edrealbox('Enter driver
B.A.C.',jr,6,3,17);
                           while not eof(datafile) do
                           begin
                                read(datafile,data);
                                case action of
                                     1: begin
                                             if
data.reals[id]=jr then
                                             begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                             end;
                                        end;
                                     2: begin
                                             if
data.reals[id]<=jr then
                                             begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                             end;
                                        end;
                                     3: begin
                                             if
data.reals[id]<jr then
```

```
                                        begin
addintlist(l,data.numeric[acc_no]);
inc(recordcount);
                                        end;
                                    end;
                                4: begin
                                        if
data.reals[id]>jr then
                                        begin
addintlist(l,data.numeric[acc_no]);
inc(recordcount);
                                        end;
                                    end;
                                5: begin
                                        if
data.reals[id]>=jr then
                                        begin
addintlist(l,data.numeric[acc_no]);
inc(recordcount);
                                        end;
                                    end;
                                6: begin
                                        if
data.reals[id]<>jr then
                                        begin
addintlist(l,data.numeric[acc_no]);
inc(recordcount);
                                        end;
                                    end;
                            end;
                        end;
                    end;
                8: begin
                        id:=road_name;
                        textcolor(errfore);
                        textbackground(errback);
                        errorbox(' Enter road name
',copy(space,1,29),16,shaddow);
                        js:='';
                        repeat
                        code:=edstr(24,16,js,30);
                        until code=0;
                        while not eof(datafile) do
                        begin
                            read(datafile,data);
                            case action of
                                1: begin
                    (                       if
data.strings[id]=js then
                                        begin
addintlist(l,data.numeric[acc_no]);
inc(recordcount);
```

```
                                        end;
                                    end;
                                2: begin
                                        if
data.strings[id]<=js then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                3: begin
                                        if
data.strings[id]<js then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                4: begin
                                        if
data.strings[id]>js then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                5: begin
                                        if
data.strings[id]>=js then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                6: begin
                                        if
data.strings[id]<>js then
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                end;
                            end;
                        end;
                    9: begin
                            id:=intersecting_road;
                            textcolor(errfore);
                            textbackground(errback);
                            errorbox(' Enter intersecting
road ',copy(space,1,29),17,shaddow);
                            js:='';
```

```
                              repeat
                              code:=edstr(24,17,js,30);
                              until code=0;
                              while not eof(datafile) do
                              begin
                                   read(datafile,data);
                                   case action of
                                        1: begin
                                                if
data.strings[id]=js then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        2: begin
                                                if
data.strings[id]<=js then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        3: begin
                                                if
data.strings[id]<js then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        4: begin
                                                if
data.strings[id]>js then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        5: begin
                                                if
data.strings[id]>=js then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        6: begin
                                                if
data.strings[id]<>js then
                                                begin
```

```
addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                            end;
                                        end;
                                    end;
                                end;

                    end;

{                    while not eof(datafile) do
                                begin
                                    read(datafile,data);
                                    case action of
                                        0: begin
                                                mainscreen;
                                                exit;
                                           end;
                                        1: begin
                                                if
data.numeric[id]=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        2: begin
                                                if
data.numeric[id]<=j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        3: begin
                                                if
data.numeric[id]<j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        4: begin
                                                if
data.numeric[id]>j then
                                                begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                                end;
                                           end;
                                        5: begin
                                                if
data.numeric[id]>=j then
```

```
                                        begin

addintlist(l,data.numeric[acc_no]);

inc(recordcount);
                                        end;
                                    end;
                                6: begin
                                        if
data.numeric[id]<>j then
                                        begin

└┘ΦΨΣ◘•••••••••±┴10@►00@Ç8p811└1└►►◘ÉAÇ◘┴üÇH↓••••••
          ••••••••••••••••••••••••••••••••••••••••••••••••••≡@◄êpæ☺Çæ1Ç
          ►8H↓•••••••••••••••••••••••••••••••••••••••••••••••••••••±◄8↓
          ••••••••••••••••••••••••••••••••••••••••••••±◄8↓••••••••••
          ••••••••••••••••••••••••±◄8↓•••••••••••••••••••••••••••••
          ±◄8▲↓•••••••••••••••••≡1p1Ç0ÿp☺1►e);
                                textbackground(modback);
                                windowbox(x1,y1,x2,y2);
                                gotoxy(1,1);
                                writelist(l);
                                windowbox(5,19,75,24);
                                gotoxy(1,1);
                                str(recordcount,s);
                                centrestr(concat('The Total number of
          records found with'),1);

          centrestr(concat('"',getchoice(menusfile,30,menuchoice),'"
          2);

          centrestr(concat(copy(getchoice(menusfile,31,action),3,len
          (getchoice(menusfile,31,action))),'      '),3);
                                gotoxy(wherex-4,wherey);
                                case choice of
                                     1: write(j);
                                     2: writedate(j);
                                     3: write(j);
                                     4: write(j);
                                     5: write(j);
                                     6: write(j);
                                     7: write(jr:0:3);
                                     8: write(js);
                                     9: write(js);
                                end;
                                writeln;
                                str(recordcount,s);
                                centrestr(concat(' is ',s),4);
                                window(1,1,80,25);
                                centrestr(' Press ESC for main menu. ',2
        getesc;
                                textcolor(mainfore);
                                textbackground(mainback);
                           end;
               end;
               mainscreen;
          {     newfile(j,temp,field,menuchoice,fieldchoice);
               mainscreen;}
          end;

          procedure show_acc_numbers;
          var
```

```
   n,i:longint;
   l:intlistptr;

procedure traverse(t:treeptr1);
begin
     if t<>nil then
     with t^ do
     begin
          traverse(left);
          addintlist(l,key);
          traverse(right);
     end;
end;

begin
     textcolor(modfore);
     textbackground(modback);
     linecount:=1;
     count:=1;
     l:=nil;
     traverse(maintree);
     writeln;
     windowbox(x1,y1,x2,y2);
     gotoxy(1,1);
     writelist(l);
     window(1,1,80,25);
     centrestr(' Press ESC for main menu. ',18);
     getesc;
     textcolor(mainfore);
     textbackground(mainback);
     mainscreen;
end;

procedure quit;
var
   yes:boolean;
begin
     yn:=true;
     error(' Verify ',' Are you sure ? (Y or N)
',boxline-1,yn);
     if yn then halt;
end;

procedure initialize;
var
   f:text;
begin
     if copyrightcheck then
     begin
          assign(f,'copyright');
          {$I-}
          reset(f);
          {$I+}
          if ioresult<>0 then
          begin
               writeln;
               writeln('Copyright violation. Program
halted.');
               writeln;
               halt(1);
          end;
     end;
```

```
        cursor(off);
        assign(listfile,menusfile);
        assign(datafile,datafilename);
        maintree:=nil;
        menustree:=nil;
        helptree:=nil;
        setbeep(mute);
        readcolors;
        readtrees;
        id:=1;
        finished:=false;
        yn:=true;
end;

end.
```

```
unit udfs3;

interface

uses
    crt,dos,udfs1;

const
     space='                                                    ';
     title='Copyright (c) 1990. By Gerard Read.';
     menusfile='menus.dat';
     helpfile='help.dat';
     datafilename='datafile.dat';
     tempfile='tempfile.dat';
     colorfile='color.dat';
     comproc='\command.com';
     path='';
     nohelp=true;
     copyrightcheck=false;

     acc_no_length=6;
     showdata=true;
     hidedata=false;
     bunch=false;
     x=5;
     xleft=37;
     xright=xleft;
     starty=7;
     boxline=22;
     x1=22;
     y1=5;
     x2=57;
     y2=20;
     pg1=true;
     pg2=false;
     pgup=#73;
     pgdn=#81;
     esc=#27;
     return=#13;
     f1=#59;
     f2=#60;
     f3=#61;
     f4=#62;
     f5=#63;
     f6=#64;
     f10=#68;

     max_acc_no=9999;
     acc_no=1;
     day_of_acc=3;
     date_of_acc=2;
     hour_of_acc=4;
     cont_circ=5;
     light=6;
     atmos_cond=7;
     road_surf=8;
     horiz_feat=9;
     vert_feat=10;
     traf_cont=11;
     special_feat=12;
     divided_road=13;
     speed_limit=14;
```

```
      unit_inv=15;
      ped_age=16;
      ped_sex=17;
      ped_drinking=18;
      vehicles_intent=19;
      severity_of_injuries=20;
      driver_age=21;
      driver_sex=22;
      driver_drinking=23;
      driver_lic=24;
      road_user=25;
      driver_bac=26;
      road_name=27;
      intersecting_road=28;

      no_of_num_fields=25;
      no_of_real_fields=1;
      no_of_str_fields=2;

no_of_fields=no_of_num_fields+no_of_str_fields+no_of_real_
lds;

      max_options=7;

type
    fieldtype=0..no_of_fields;
    ntype=0..max_options;
    string50=string[50];
    stringtype=string;
    datatype=longint;
    recordtype=record
                        numeric:array[1..no_of_num_fields] of
datatype;

reals:array[no_of_num_fields+1..no_of_real_fields+no_of_nu
ields] of real;

strings:array[no_of_num_fields+no_of_real_fields+1..no_of_
lds] of stringtype;
                    end;
    filetype=file of recordtype;

var
   listfile:text;
   key:char;
   maintree:treeptr1;
   menustree,helptree:treeptr2;
   data:recordtype;
   temp,datafile:filetype;

errfore,errback,mainfore,mainback,helpfore,helpback,helpbo
er,
   modfore,modback,high,bar,mfore,mback,mhigh,mbar,mborder
   mbfore,mbback,mbhigh,mbbar:integer;
   list:dat;
   bchoice,choice,id,bn,n,i:integer;

yn,helpfilenotfound,datafilenotfound,keyexists,finished,dr
rdrinking:boolean;
   linecount,count,code,recordcount,counter:integer;

procedure addintlist(var l:intlistptr;n:longint);
```

```
procedure dohelp(id:integer);
procedure writerecord(rec:recordtype);
procedure writescreen(showdata,firstpage:boolean);
procedure boxit;
procedure setcolors;
procedure writelist(l:intlistptr);
procedure get(no:integer;var c:integer);
procedure printn(x,y:integer;s:string);
procedure printh(x,y:integer;s:string);
procedure readcolors;
procedure changecolors(color:integer);
procedure shell;
procedure readdate(var date:datatype;var code:integer);
procedure writedate(date:longint);
procedure edintbox(a:string;var i:longint;len,line:integer
procedure edrealbox(a:string;var
i:real;len,dec,line:integer);
procedure editreal(x,y:integer;var r:real;a,b,max:integer)
procedure error(a,b:string;line:integer;var k:boolean);
procedure getyn(var y:boolean);
procedure getlist(filename:string;no:integer);
procedure addstrlist(var l:strlistptr;n:string);
function getday(y,m,d:word;var
code:integer;today:boolean):word;
function getchoice(filename:string;a,b:integer):string;
function searchlist(ls:strlistptr;n:integer):string;

implementation

procedure addintlist(var l:intlistptr;n:longint);
var
   p,q:intlistptr;
begin
     new(p);
     p^.info:=n;
     p^.next:=nil;
     if l=nil then
        l:=p
     else
     begin
          q:=l;
          while q^.next<>nil do
                q:=q^.next;
          q^.next:=p;
     end;
end;

procedure addstrlist(var l:strlistptr;n:string);
var
   p,q:strlistptr;
begin
     new(p);
     p^.info:=n;
     p^.next:=nil;
     if l=nil then
        l:=p
     else
     begin
          q:=l;
          while q^.next<>nil do
                q:=q^.next;
          q^.next:=p;
```

```
     end;
end;

procedure getlist(filename:string;no:integer);
var
   lst:strlistptr;
   line:string;
   i,code,k,maxlen,recno:integer;
   nofound,readok:boolean;
   treename:treeptr2;

procedure readlist(ls:strlistptr);
var
   l:strlistptr;
begin
     l:=ls;
     k:=1;
     while l<>nil do
     begin
          list[k]:=l^.info;
          l:=l^.next;
          maxlen:=max(length(list[k]),maxlen);
          inc(k);
     end;
end;

begin
     if filename=menusfile then
        treename:=menustree;
     if filename=helpfile then
        treename:=helptree;
     maxlen:=0;
     lst:=search2(treename,no);
     readlist(lst);
     n:=k-1;
     readok:=true;
     for i:=1 to n do
         list[i]:=concat('
',list[i],copy(space,1,maxlen-length(list[i])),' ');
end;

function searchlist(ls:strlistptr;n:integer):string;
var
   l:strlistptr;
   i:integer;
begin
     l:=ls;
     for i:=1 to n-1 do
     begin
          l:=l^.next;
     end;
     searchlist:=l^.info;
end;

function getchoice(filename:string;a,b:integer):string;
var
   line:string;
   i,code,k,maxlen:integer;
   nofound:boolean;
   lst:strlistptr;
begin
     lst:=search2(menustree,a);
```

```
     getchoice:=searchlist(lst,b);
     nofound:=true;
     if ioresult<>0 then
end;


procedure getyn(var y:boolean);
var
   key:char;
   finished:boolean;
begin
     finished:=false;
     repeat
          key:=readkey;
          case key of
               'y','Y': begin
                             y:=true;
                             finished:=true;
                        end;
               'n','N': begin
                             y:=false;
                             finished:=true;
                        end;
               f1: dohelp(1);
          else
              beep(high);
          end;
     until finished;
end;

procedure error(a,b:string;line:integer;var k:boolean);
begin
     beep(med);
     screen(save);
     textcolor(errfore);
     textbackground(errback);
     errorbox(a,b,line,shaddow);
     if k then
        getyn(k)
     else
         getesc;
     screen(restore);
     if ioresult<>0 then
     textcolor(modfore);
     textbackground(modback);
end;

procedure editreal(x,y:integer;var r:real;a,b,max:integer)
var
   code,c:integer;
   oldr:real;
begin
     cursor(small);
     oldr:=r;
     repeat
          code:=edreal(x,y,r,a,b,max,c);
          case code of
               27: r:=oldr;
               59: dohelp(2);
          end;
     until code=0;
     cursor(off);
```

```
end;

procedure edintbox(a:string;var i:longint;len,line:integer
var
   code:integer;
begin
     cursor(small);
     screen(save);
     textcolor(errfore);
     textbackground(errback);
     editbox(a,i,len,line,code);
     screen(restore);
     cursor(off);
end;

procedure edrealbox(a:string;var
i:real;len,dec,line:integer);
begin
     cursor(small);
     screen(save);
     textcolor(errfore);
     textbackground(errback);
     editrbox(a,i,len,dec,line,code);
     screen(restore);
     cursor(off);
end;

function getday(y,m,d:word;var
code:integer;today:boolean):word;
var
   gd,h,i,j,k:word;
begin
     getdate(h,i,j,k);
     setdate(y+1900,m,d);
     getdate(y,m,d,gd);
     getday:=gd+1;
     if (y=h) and (m=i) and (d=j) and (not today) then
        code:=1
     else
         code:=0;
     setdate(h,i,j);
     if today then code:=0;
end;

procedure writedate(date:longint);
var
   s:string;
begin
     str(date,s);
     write(s[5],s[6],'/',s[3],s[4],'/',s[1],s[2]);
end;

procedure readdate(var date:datatype;var code:integer);

procedure editdate(var date:datatype);
var
   a,b,c:datatype;
   x,y,code:integer;
begin
     cursor(small);
     finished:=false;
     yn:=false;
```

```
      x:=wherex;
      y:=wherey;
      repeat
            edint(x,y,a,2,code);
            if code=59 then
            begin
                 dohelp(3);
                 finished:=false;
            end;
            if (a<1) or (a>31) then
            begin
                 finished:=false;
                 error(' Error ',' Invalid day of month. Pr
ESC. ',12,yn);
                 gotoxy(x,y);
                 write('  ');
                 gotoxy(x,y);
            end
            else
                finished:=true;
      until finished;
      write('/');
      finished:=false;
      x:=wherex;
      y:=wherey;
      repeat
            edint(x,y,b,2,code);
            if code=59 then
            begin
                 dohelp(4);
                 finished:=false;
            end;
            if (b<1) or (b>12) then
            begin
                 finished:=false;
                 error(' Error ',' Invalid month. Press ESC
',12,yn);
                 gotoxy(x,y);
                 write('  ');
                 gotoxy(x,y);
            end
            else
                finished:=true;
      until finished;
      write('/');
      edint(wherex,wherey,c,2,code);
      if code=59 then
            begin
                 dohelp(5);
                 finished:=false;
            end;
      date:=a+100*b+10000*c;
      cursor(off);
end;

var
   year,month,day,dayofweek,
   y,m,d,dw,
   oy,om,od,odw:word;
   today:boolean;
begin
      editdate(date);
```

```
     y:=trunc(date/10000);
     m:=trunc((date-(10000*trunc(date/10000)))/100);
     d:=date-(trunc(date/100)*100);
     getdate(year,month,day,dayofweek);
     if (y=year-1900) and (m=month) and (d=day) then
        today:=true
     else
         today:=false;
     dw:=getday(y,m,d,code,today);
     if code<>0 then exit;
     if not today then
     begin

     end;
     data.numeric[day_of_acc]:=dw;
end;

procedure shell;
begin
     cursor(small);
     textcolor(7);
     textbackground(0);
     screen(save);
     clrscr;
     writeln('Type EXIT to return to Pedestrian Accident
System...');
     exec(concat(path,comproc),'');
     if doserror<>0 then
     begin
          writeln;
          writeln('Shell error no ',doserror,'. Press any
key...');
          repeat until keypressed;
     end;
     screen(restore);
     textcolor(mainfore);
     textbackground(mainback);
     cursor(off);
end;

procedure changecolors(color:integer);
var
   f:text;
   fname:string;
begin
     case color of
          2: begin
                  fname:='color.1';
             end;
          3: begin
                  fname:='color.2';
             end;
          4: begin
                  fname:='color.3';
             end;
          else
              begin
                   errfore:=15;
                   errback:=0;
                   mainfore:=7;
                   mainback:=0;
                   helpfore:=0;
```

```
                    helpback:=7;
                    helpboarder:=2;
                    modfore:=7;
                    modback:=0;
                    high:=15;
                    bar:=7;
                    mfore:=15;
                    mback:=7;
                    mhigh:=15;
                    mbar:=0;
                    mborder:=0;
                end;
            end;
        case color of
            2,3,4: begin
                        assign(f,fname);
                        {$I-}
                        reset(f);
                        {$I+}
                        if ioresult<>0 then
                        begin
                            yn:=false;
                            error(' Error ',concat(' File
',fname,' not found. Press ESC. '),12,yn);
                        end
                        else
                        begin
                            readln(f,errfore);
                            readln(f,errback);
                            readln(f,mainfore);
                            readln(f,mainback);
                            readln(f,helpfore);
                            readln(f,helpback);
                            readln(f,helpboarder);
                            readln(f,modfore);
                            readln(f,modback);
                            readln(f,high);
                            readln(f,bar);
                            readln(f,mfore);
                            readln(f,mback);
                            readln(f,mhigh);
                            readln(f,mbar);
                            readln(f,mborder);
                        end;
                    end;
        end;
        assign(f,colorfile);
        rewrite(f);
        write(f,color);
        close(f);
        seteditcolor(high,bar);
        setmenubarcolor(1,mbfore,mbback,mbbar,mbhigh);
        for i:=1 to max_id+1 do
            setmenucolor(i,mfore,mback,mhigh,mbar,mborder);
end;

procedure readcolors;
var
   f:text;
   code,color:integer;
   line:string;
begin
```

```
     assign(f,colorfile);
     {$I-}
     reset(f);
     {$I+}
     if ioresult<>0 then
     begin
          textbackground(0);
          clrscr;
          color:=1;
          changecolors(color);
          yn:=false;
          error(' Error ',concat('File ',colorfile,' not
found. Press ESC. '),12,yn);
     end
     else
     begin
          readln(f,line);
          val(line,color,code);
          if code<>0 then
          begin
               color:=1;
               changecolors(color);
               yn:=false;
               error(' Error ',' Error in Addpedac.col fil
Contact programmer. Press ESC. ',12,yn);
          end
          else
              changecolors(color);
     end;

end;

procedure printh(x,y:integer;s:string);

begin
     textcolor(high);
     textbackground(bar);
     print(x-length(s),y,s);
     textcolor(modfore);
     textbackground(modback);
end;

procedure printn(x,y:integer;s:string);
begin
     textcolor(modfore);
     textbackground(modback);
     print(x-length(s),y,s);
end;

procedure get(no:integer;var c:integer);
var
   xpos,ypos,x,y:integer;
begin
     if no<=14 then
     begin
          xpos:=48;
          ypos:=6;
     end;
     if no>=15 then
     begin
          xpos:=48;
          ypos:=6;
```

```
      end;
      case no of
           1: begin
                    printh(xleft,starty,'Accident Number: ')
                    repeat

 edint(wherex,wherey,data.numeric[no],acc_no_length,code);
                         if code=27 then
                         begin
                              c:=-1;
                              exit;
                         end
                         else
                             c:=0;
                         if code=59 then
                         begin
                              dohelp(6);
                              finished:=false;
                         end;
                    until code=0;
                    printn(xleft,starty,'Accident Number: ')
                    write(data.numeric[no]:acc_no_length);
                    exit;
               end;
           2: begin
                    finished:=false;
                    yn:=false;
                    repeat
                         printh(xleft,starty+1,'Date of
 Accident: ');
                         readdate(data.numeric[no],code);
                         if code<>0 then
                         begin
                              finished:=false;
                              error(' Error ',' Invalid dat
 Press ESC. ',12,yn);
                         end
                         else
                         finished:=true;
                    until finished;
                    printn(xleft,starty+1,'Date of Accident:
 ');
                    writedate(data.numeric[no]);
                    exit;
               end;
           3: begin
                    printh(xleft,starty+2,'Day of Accident:

 write(getchoice(menusfile,day_of_acc,data.numeric[no]));
                    printn(xleft,starty+2,'Day of Accident:
exit;
               end;
           4: begin
                    finished:=false;
                    yn:=false;
                    repeat
                         printh(xleft,starty+3,'Hour of
 Accident: ');
                         repeat

 edint(wherex,wherey,data.numeric[no],2,code);
                              if code=59 then
```

```
                                    begin
                                         dohelp(7);
                                         finished:=false;
                                    end;
                               until code=0;
                               printn(xleft,starty+3,'Hour of
Accident: ');
                               write(data.numeric[no]:2);
                               if data.numeric[no]>24 then
                               begin
                                    error(' Error ',' Hour not
valid. Press ESC. ',12,yn);
                                    finished:=false;
                               end
                               else
                                   finished:=true;
                          until finished;
                          exit;
                     end;
                5: begin
                          printh(xleft,starty+4,'Contributing
Circumstances: ');
                          getlist(menusfile,cont_circ);
                     end;
                6: begin
                          printh(xleft,starty+5,'Light: ');
                          getlist(menusfile,light);
                     end;
                7: begin
                          printh(xleft,starty+6,'Atmospheric
Conditions: ');
                          getlist(menusfile,atmos_cond);
                     end;
                8: begin
                          printh(xleft,starty+7,'Road Surface: ');
                          getlist(menusfile,road_surf);
                     end;
                9: begin
                          printh(xleft,starty+8,'Horizontal Featur
');
                          getlist(menusfile,horiz_feat);
                     end;
               10: begin
                          printh(xleft,starty+9,'Vertical Feature
');
                          getlist(menusfile,vert_feat);
                     end;
               11: begin
                          printh(xleft,starty+10,'Traffic Control
');
                          getlist(menusfile,traf_cont);
                          xpos:=39;
                     end;
               12: begin
                          printh(xleft,starty+11,'Special Feature
');
                          getlist(menusfile,special_feat);
                          xpos:=45;
                          ypos:=6;
                     end;
               13: begin
                          printh(xleft,starty+12,'Divided Road: '
```

```
                    getlist(menusfile,divided_road);
               end;
          14: begin
                    printh(xleft,starty+13,'Speed Limit: ')
                    x:=wherex;
                    y:=wherey;
                    repeat
                         edint(x,y,data.numeric[no],3,code
                         if code=59 then
                         begin
                              dohelp(5);
                              finished:=false;
                         end;
                         if data.numeric[no]<1 then
                         begin
                              yn:=false;
                              code:=-1;
                              error(' Error ',' Invalid sp
limit. Press ESC. ',12,yn);
                         end;
                    until code=0;
                    printn(xleft,starty+13,'Speed Limit: ')
                    exit;
               end;
          15: begin
                    printh(xright,starty,'Unit Involved: ')
                    getlist(menusfile,unit_inv);
               end;
          16: begin
                    printh(xright,starty+1,'Pedestrian Age:
');
                    x:=wherex;
                    y:=wherey;
                    repeat
                         edint(x,y,data.numeric[no],3,code
                         if code=59 then
                         begin
                              dohelp(6);
                              finished:=false;
                         end;
                         if data.numeric[no]<1 then
                         begin
                              yn:=false;
                              code:=-1;
                              error(' Error ',' Invalid
pedestrian age. Press ESC. ',12,yn);
                         end;
                    until code=0;
                    printn(xright,starty+1,'Pedestrian Age:
');
                    write(data.numeric[no]:3);
                    exit;
               end;
          17: begin
                    printh(xright,starty+2,'Pedestrian Sex:
');
                    getlist(menusfile,ped_sex);
               end;
          18: begin
                    printh(xright,starty+3,'Pedestrian
Drinking: ');
                    getlist(menusfile,ped_drinking);
```

```
                end;
            19: begin
                    printh(xright,starty+4,'Vehicles
Intention: ');
                    getlist(menusfile,vehicles_intent);
                end;
            20: begin
                    Printh(xright,starty+5,'Severity of
Injuries: ');
                    getlist(menusfile,severity_of_injuries)
                end;
            21: begin
                    Printh(xright,starty+6,'Driver Age: ');
                    x:=wherex;
                    y:=wherey;
                    repeat
                          edint(x,y,data.numeric[no],3,code
                          if code=59 then
                          begin
                               dohelp(6);
                               finished:=false;
                          end;
                          if data.numeric[no]<17 then
                          begin
                               yn:=false;
                               code:=-1;
                               error(' Error ',' Invalid
driver age. Press ESC. ',12,yn);
                          end;
                    until code=0;
                    Printn(xright,starty+6,'Driver Age: ');
                    write(data.numeric[no]:3);
                    exit;
                end;
            22: begin
                    Printh(xright,starty+7,'Driver Sex: ');
                    getlist(menusfile,driver_sex);
                end;
            23: begin
                    Printh(xright,starty+8,'Driver Drinking
');
                    getlist(menusfile,driver_drinking);
                end;
            24: begin
                    Printh(xright,starty+9,'Drivers Licence
');
                    getlist(menusfile,driver_lic);
                end;
            25: begin
                    Printh(xright,starty+10,'Road User: ');
                    getlist(menusfile,road_user);
                end;
            26: begin
                    Printh(xright,starty+11,'Driver B.A.C.:
');
                    if not driverdrinking then
                       data.reals[no]:=0.0
                    else

editreal(wherex,wherey,data.reals[no],1,2,4);
                    printn(xright,starty+11,'Driver B.A.C.:
');
```

```
                    write(data.reals[no]:0:3);
                    exit;
               end;
          27: begin
                    printh(xright,starty+12,'Road Name: ');
                    data.strings[no]:='';
                    repeat

code:=edstr(wherex,wherey,data.strings[no],30);
                         case code of
                              59: dohelp(1);
                         end;
                    until code=0;
                    printn(xright,starty+12,'Road Name: ');
                    write(data.strings[no]);

write(copy(space,1,30-length(data.strings[no])));
                    exit;
               end;
          28: begin
                    printh(xright,starty+13,'Intersecting
Road: ');
                    data.strings[no]:='';
                    repeat

code:=edstr(wherex,wherey,data.strings[no],30);
                         case code of
                              59: dohelp(1);
                         end;
                    until code=0;
                    printn(xright,starty+13,'Intersecting
Road: ');
                    write(data.strings[no]);

write(copy(space,1,30-length(data.strings[no])));
                    exit;
               end;
     end;

     screen(save);
     menu(no,xpos,ypos,list,n,choice,false);
     screen(restore);

     case no of
          5: begin
                    printn(xleft,starty+4,'Contributing
Circumstances: ');

write(getchoice(menusfile,cont_circ,choice));
               end;
          6: begin
                    printn(xleft,starty+5,'Light: ');
                    write(getchoice(menusfile,light,choice))
               end;
          7: begin
                    printn(xleft,starty+6,'Atmospheric
Conditions: ');

write(getchoice(menusfile,atmos_cond,choice));
               end;
          8: begin
                    printn(xleft,starty+7,'Road Surface: ');
```

```
write(getchoice(menusfile,road_surf,choice));
                end;
            9: begin
                    printn(xleft,starty+8,'Horizontal Featur
');

write(getchoice(menusfile,horiz_feat,choice));
                end;
           10: begin
                    printn(xleft,starty+9,'Vertical Feature
');

write(getchoice(menusfile,vert_feat,choice));
                end;
           11: begin
                    printn(xleft,starty+10,'Traffic Control
');

write(getchoice(menusfile,traf_cont,choice));
                end;
           12: begin
                    printn(xleft,starty+11,'Special Feature
');

write(getchoice(menusfile,special_feat,choice));
                end;
           13: begin
                    printn(xleft,starty+12,'Divided Road: '

write(getchoice(menusfile,divided_road,choice));
                end;
           14: begin
                    printn(xleft,starty+13,'Speed Limit: ')

write(getchoice(menusfile,speed_limit,choice));
                end;
           15: begin
                    printn(xright,starty,'Unit Involved: ')

write(getchoice(menusfile,unit_inv,choice));
                end;
           16: begin
                    printn(xright,starty+1,'Pedestrian Age:
');
                    write(data.numeric[ped_age]);
                end;
           17: begin
                    printn(xright,starty+2,'Pedestrian Sex:
');

write(getchoice(menusfile,ped_sex,choice));
                end;
           18: begin
                    printn(xright,starty+3,'Pedestrian
Drinking: ');

write(getchoice(menusfile,ped_drinking,choice));
                end;
           19: begin
                    printn(xright,starty+4,'Vehicles
Intention: ');
```

```
write(getchoice(menusfile,vehicles_intent,choice));
                end;
           20: begin
                     printn(xright,starty+5,'Severity of
Injuries: ');

write(getchoice(menusfile,severity_of_injuries,choice));
                end;
           21: begin
                     printn(xright,starty+6,'Driver Age: ');

write(getchoice(menusfile,driver_age,choice));
                end;
           22: begin
                     printn(xright,starty+7,'Driver Sex: ');

write(getchoice(menusfile,driver_sex,choice));
                end;
           23: begin
                     printn(xright,starty+8,'Driver Drinking
');

write(getchoice(menusfile,driver_drinking,choice));
                     if choice=1 then
                        driverdrinking:=true
                     else
                         driverdrinking:=false;
                end;
           24: begin
                     printn(xright,starty+9,'Drivers Licence
');

write(getchoice(menusfile,driver_lic,choice));
                end;
           25: begin
                     printn(xright,starty+10,'Road User: ');

write(getchoice(menusfile,road_user,choice));
                end;
{           26: begin
                     if not driverdrinking then
data.reals[no]:=0.0;
                     printn(xright,starty+11,'Driver B.A.C.:
');
                     write(data.reals[no]);
                end;}
           27: begin
                     printn(xright,starty+12,'Road Name: ');
                     write(data.strings[no]);
                end;
           28: begin
                     printn(xright,starty+12,'Intersecting
Road: ');
                     write(data.strings[no]);
                end;
     end;
     data.numeric[no]:=choice;
end;

procedure writelist(l:intlistptr);
const
```

```
     space='                              ';
     across=5;
     down=11;
var
   s:string;
   a,b,c,d,e:integer;
begin
     if l<>nil then
     begin
          if linecount>down then
          begin
               getwindow(a,b,c,d);
               e:=wherey;
               screen(save);
               window(1,1,80,25);
               centrestr(' Press any key... ',18);
               keyhalt;
               screen(restore);
               window(a,b,c,d);
               clrscr;
               gotoxy(1,1);
               linecount:=1;
          end;
          str(l^.info,s);
          write(s,copy(space,1,7-length(s)));
          if ((count div across)=(count/across)) and
(count<>0) then
          begin
{              writeln;}
               inc(linecount);
          end;
          inc(count);
          writelist(l^.next);
     end;
end;



procedure writerecord(rec:recordtype);
begin
     write(datafile,rec);
end;

procedure writescreen(showdata,firstpage:boolean);
var
   a:byte;
   x1,y1,x2,y2:integer;

procedure boxit;
begin
     dbox(x1,y1,x2,y2);
     if showdata then
        centrestr(' V i e w   A c c i d e n t   D a t a
',y1)
     else
         centrestr(' A d d   D a t a  ',y1);
end;

 begin
      textcolor(modfore);
      textbackground(modback);
```

```
        x1:=7;
        y1:=starty-2;
        x2:=screenwidth-x1*2+1;
        y2:=18;
        window(x1,y1,80-x1+1,y2+6);
        clrscr;
        window(1,1,80,25);
        if showdata then
        begin
             with data do
             begin
                  if firstpage then
                  begin
                       counter:=starty;
                       boxit;
                       centrestr('Page ONE.',boxline);
                       centrestr('PgDn for more or ESC to
Quit.',boxline+1);
                       printn(xleft,starty,'Accident Number:
');
                       write(numeric[acc_no]);
                       printn(xleft,starty+1,'Date of Accide
');
                       writedate(numeric[date_of_Acc]);
                       printn(xleft,starty+2,'Day of Acciden
');

write(getchoice(menusfile,day_of_acc,numeric[day_of_acc]))
                       printn(xleft,starty+3,'Hour of Accide
');
                       write(numeric[hour_of_acc]);
                       printn(xleft,starty+4,'Contributing
Circumstances: ');

write(getchoice(menusfile,cont_circ,numeric[cont_circ]));
                       printn(xleft,starty+5,'Light: ');

write(getchoice(menusfile,light,numeric[light]));
                       printn(xleft,starty+6,'Atmospheric
Conditions: ');

write(getchoice(menusfile,atmos_cond,numeric[atmos_cond]))
                       printn(xleft,starty+7,'Road Surface:

write(getchoice(menusfile,road_surf,numeric[road_surf]));
                       printn(xleft,starty+8,'Horizontal
Features: ');

write(getchoice(menusfile,horiz_feat,numeric[horiz_feat]))
                       printn(xleft,starty+9,'Vertical
Features: ');

write(getchoice(menusfile,vert_feat,numeric[vert_feat]));
                       printn(xleft,starty+10,'Trafic Contro
');

write(getchoice(menusfile,traf_cont,numeric[traf_cont]));
                       printn(xleft,starty+11,'Special
Features: ');

write(getchoice(menusfile,special_feat,numeric[special_fea
);
```

```
                        printn(xleft,starty+12,'Divided Road:
');

write(getchoice(menusfile,divided_road,numeric[divided_roa
);
                        printn(xleft,starty+13,'Speed Limit:
  write(numeric[speed_limit]);
                end
                else
                begin
                        counter:=starty;
                        boxit;
                        centrestr('Page TWO.',boxline);
                        centrestr('PgUn for more or ESC to
Quit.',boxline+1);
                        printn(xright,starty,'Unit Involved:

write(getchoice(menusfile,unit_inv,numeric[unit_inv]));
                        printn(xright,starty+1,'Pedestrian Ag
');
                        write(numeric[ped_age]);
                        printn(xright,starty+2,'Pedesrtian Se
');

write(getchoice(menusfile,ped_sex,numeric[ped_sex]));
                        printn(xright,starty+3,'Pedestrian
Drinking: ');

write(getchoice(menusfile,ped_drinking,numeric[ped_drinkin
);
                        printn(xright,starty+4,'Vehicles
Intention: ');

write(getchoice(menusfile,vehicles_intent,numeric[vehicles
tent]));
                        printn(xright,starty+5,'Severity of
Injuries: ');

write(getchoice(menusfile,severity_of_injuries,numeric[sev
ty_of_injuries]));
                        printn(xright,starty+6,'Driver Age: '
                        write(numeric[driver_age]);
                        printn(xright,starty+7,'Driver Sex: '

write(getchoice(menusfile,driver_sex,numeric[driver_sex]))
                        printn(xright,starty+8,'Driver Drinki
');

write(getchoice(menusfile,driver_drinking,numeric[driver_d
king]));
                        printn(xright,starty+9,'Drivers Licen
');

write(getchoice(menusfile,driver_lic,numeric[driver_lic]))
                        printn(xright,starty+10,'Road User: '

write(getchoice(menusfile,road_user,numeric[road_user]));
                        printn(xright,starty+11,'Driver B.A.C
');
                        write(reals[driver_bac]:0:2);
                        printn(xright,starty+12,'Road Name: '
                        write(strings[road_name]);
```

```
                    printn(xright,starty+13,'Intersecting
Road: ');
                    write(strings[intersecting_road]);
               end;
          end;
     end
     else
     begin
          if firstpage then
          begin
               counter:=starty;
               boxit;
               centrestr('Page ONE.',boxline);
               printn(xleft,starty,'Accident Number: ');
               printn(xleft,starty+1,'Date of Accident: ')
               printn(xleft,starty+2,'Day of Accident: ');
               printn(xleft,starty+3,'Hour of Accident: ')
               printn(xleft,starty+4,'Contributing
Circumstances: ');
               printn(xleft,starty+5,'Light: ');
               printn(xleft,starty+6,'Atmospheric Conditio
');
               printn(xleft,starty+7,'Road Surface: ');
               printn(xleft,starty+8,'Horizontal Features:
');
               printn(xleft,starty+9,'Vertical Features: '
               printn(xleft,starty+10,'Trafic Control: ');
               printn(xleft,starty+11,'Special Features: '
               printn(xleft,starty+12,'Divided Road: ');
               printn(xleft,starty+13,'Speed Limit: ');
          end
          else
          begin
               counter:=starty;
               boxit;
               centrestr('Page TWO.',boxline);
               printn(xright,starty,'Unit Involved: ');
               printn(xright,starty+1,'Pedestrian Age: ');
               printn(xright,starty+2,'Pedesrtian Sex: ');
               printn(xright,starty+3,'Pedestrian Drinking
');
               printn(xright,starty+4,'Vehicles Intention:
');
               printn(xright,starty+5,'Severity of Injurie
');
               printn(xright,starty+6,'Driver Age: ');
               printn(xright,starty+7,'Driver Sex: ');
               printn(xright,starty+8,'Driver Drinking: ')
               printn(xright,starty+9,'Drivers Licence: ')
               printn(xright,starty+10,'Road User: ');
               printn(xright,starty+11,'Driver B.A.C.: ');
               printn(xright,starty+12,'Road Name: ');
               printn(xright,starty+13,'Intersecting Road:
');
          end;
     end;
end;

procedure dohelp(id:integer);
var
   x,y:integer;
begin
```

```
     if nohelp then exit;
     if helpfilenotfound then
     begin
          yn:=false;
          error(' Error ',concat('File ',helpfile,' not
found. Contact programmer. Press ESC.'),12,yn);
          exit;
     end;
     x:=auto;
     y:=auto;
     textcolor(helpfore);
     textbackground(helpback);
     screen(save);
     getlist(helpfile,id);
     helpbox(1,helpfore,helpback,helpboarder,n,list,x,y);
     screen(restore);
end;

procedure setcolors;
begin
     getlist(menusfile,27);
     menu(27,58,14,list,n,choice,false);
     changecolors(choice);
end;

procedure boxit;
begin
     dbox(x1,y1,x2,y2);
     if showdata then
        centrestr(' V i e w   A c c i d e n t   D a t a
',y1)
     else
         centrestr(' A d d   D a t a  ',y1);
end;

end.
```

APPENDIX B

B.1 Survey Forms

Attached are a copy of the survey forms discussed in section 5 of the report. The first is the student survey form, followed by the adult reply form.

B.2 Responses

The students responses (yearly breakdown) for Questions 1, 1a, 2, 3a, 3b are given in that order down the page following the two survey forms.

B.3 Behaviour Indices

The detailed results of the pedestrian behaviour observations for the signallised and uncontrolled crossings follow the responses.

PEDESTRIAN QUESTIONNAIRE

U.C.C.Q. 1990 GWR

Please tick the appropriate box.

Year Level 3 [] 4 [] 5 [] 6 [] 7 []
8 [] 9 [] 10 [] 11 [] 12 []

Sex MALE [] FEMALE []

1. How often do you walk to school ?

3 DAYS/WK [] RARELY []
4 DAYS/WK [] NEVER []
5 DAYS/WK []

a) If you ticked 3 DAYS/WK, 4 DAYS/WK, 5 DAYS/WK in Question 1, go to Question 2, if not how do you usually travel to school ?

BUS [] CAR []
BICYCLE [] OTHER []

2. Do you find roundabouts difficult to cross when walking ?

YES [] SOMETIMES [] NO []

3. On a scale of 1 to 5, how safe do you feel when crossing a zebra crossing ?

a) With "Lollipop" People

VERY SAFE.....1 []
- 2 []
- 3 []
- 4 []
VERY UNSAFE...5 []

b) Without "Lollipop" People

VERY SAFE.....1 []
- 2 []
- 3 []
- 4 []
VERY UNSAFE...5 []

4. Have you ever been hit by a vehicle, when walking or standing on or near the road ?

YES [] NO []

If you ticked YES, was it reported to police ?

YES [] NO []

PEDESTRIAN QUESTIONNAIRE

U.C.C.Q. 1990 GWR

Please tick the appropriate box.

Sex MALE [] FEMALE []

1. Do you find roundabouts difficult to cross when walking ?

YES [] SOMETIMES [] NO []

2. On a scale of 1 to 5, how safe do you feel when using the following crossings ?

a) Signalised control

VERY SAFE.....1 []
- 2 []
- 3 []
- 4 []
VERY UNSAFE...5 []

b) Uncontrolled Zebra

VERY SAFE.....1 []
- 2 []
- 3 []
- 4 []
VERY UNSAFE...5 []

4. Have you ever been hit by a vehicle, when walking or standing on or near the road ?

YES [] NO []

If you ticked YES, was it reported to police ?

YES [] NO []

YEAR LEVEL 4			
	M	F	TOTAL
3	0.0	3.7	1.9
4	8.7	7.4	8.1
5	39.1	40.7	39.9
RARELY	34.8	29.6	32.2
NEVER	17.4	18.5	18.0
BUS	4.3	11.1	7.7
CAR	4.3	7.4	5.9
BIKE	43.5	29.6	36.6
OTHER	0.0	0.0	0.0
YES	17.4	18.5	18.0
SOMETIMES	56.5	55.6	56.0
NO	26.1	25.9	26.0
1	52.2	63.0	57.6
2	30.4	25.9	28.2
3	8.7	7.4	8.1
4	0.0	3.7	1.9
5	8.7	0.0	4.3
1	30.4	14.8	22.6
2	17.4	22.2	19.8
3	34.8	40.7	37.8
4	4.3	14.8	9.6
5	13.0	7.4	10.2

YEAR LEVEL 5			
	M	F	TOTAL
3	8.7	11.1	9.9
4	13.0	3.7	8.4
5	43.5	44.4	44.0
RARELY	30.4	29.6	30.0
NEVER	4.3	11.1	7.7
BUS	4.3	18.5	11.4
CAR	8.7	3.7	6.2
BIKE	21.7	18.5	20.1
OTHER	0.0	0.0	0.0
YES	13.0	25.9	19.5
SOMETIMES	52.2	51.9	52.0
NO	34.8	22.2	28.5
1	52.2	70.4	61.3
2	30.4	22.2	26.3
3	17.4	0.0	8.7
4	0.0	0.0	0.0
5	0.0	3.7	1.9
1	17.4	29.6	23.5
2	34.8	14.8	24.8
3	39.1	18.5	28.8
4	0.0	14.8	7.4
5	8.7	22.2	15.5

YEAR LEVEL 6			
	M	F	TOTAL
3	3.2	3.6	3.4
4	3.2	0.0	1.6
5	32.3	25.0	28.6
RARELY	22.6	25.0	23.8
NEVER	38.7	46.4	42.6
BUS	6.5	0.0	3.2
CAR	29.0	46.4	37.7
BIKE	32.3	35.7	34.0
OTHER	0.0	0.0	0.0
YES	3.2	7.1	5.2
SOMETIMES	38.7	42.9	40.8
NO	58.1	50.0	54.0
1	64.5	64.3	64.4
2	22.6	32.1	27.4
3	0.0	3.6	1.8
4	6.5	0.0	3.2
5	6.5	0.0	3.2
1	16.1	10.7	13.4
2	25.8	7.1	16.5
3	22.6	50.0	36.3
4	22.6	10.7	16.6
5	12.9	21.4	17.2

YEAR LEVEL 7			
	M	F	TOTAL
3	3.6	6.7	5.1
4	10.7	0.0	5.4
5	10.7	6.7	8.7
RARELY	32.1	33.3	32.7
NEVER	42.9	53.3	48.1
BUS	14.3	60.0	37.1
CAR	7.1	20.0	13.6
BIKE	50.0	20.0	35.0
OTHER	3.6	0.0	1.8
YES	7.1	6.7	6.9
SOMETIMES	14.3	26.7	20.5
NO	78.6	66.7	72.6
1	42.9	46.7	44.8
2	32.1	40.0	36.1
3	14.3	13.3	13.8
4	3.6	0.0	1.8
5	7.1	0.0	3.6
1	3.6	0.0	1.8
2	7.1	0.0	3.6
3	21.4	6.7	14.0
4	32.1	33.3	32.7
5	35.7	60.0	47.9

YEAR LEVEL 8			
	M	F	TOTAL
3	0.0	0.0	0.0
4	0.0	0.0	0.0
5	25.0	15.0	20.0
RARELY	25.0	10.0	17.5
NEVER	50.0	75.0	62.5
BUS	25.0	40.0	32.5
CAR	25.0	35.0	30.0
BIKE	40.0	30.0	35.0
OTHER	0.0	0.0	0.0
YES	10.0	25.0	17.5
SOMETIMES	60.0	55.0	57.5
NO	30.0	20.0	25.0
1	55.0	55.0	55.0
2	30.0	25.0	27.5
3	10.0	10.0	10.0
4	5.0	5.0	5.0
5	0.0	5.0	2.5
1	5.0	10.0	7.5
2	30.0	20.0	25.0
3	50.0	35.0	42.5
4	10.0	20.0	15.0
5	5.0	15.0	10.0

YEAR LEVEL 9			
	M	F	TOTAL
3	0.0	0.0	0.0
4	0.0	0.0	0.0
5	5.0	18.5	11.8
RARELY	25.0	44.4	34.7
NEVER	70.0	37.0	53.5
BUS	35.0	33.3	34.2
CAR	10.0	14.8	12.4
BIKE	65.0	40.7	52.9
OTHER	0.0	0.0	0.0
YES	15.0	33.3	24.2
SOMETIMES	35.0	51.9	43.4
NO	50.0	14.8	32.4
1	65.0	25.9	45.5
2	10.0	22.2	16.1
3	20.0	3.7	11.9
4	0.0	7.4	3.7
5	5.0	3.7	4.4
1	30.0	11.1	20.6
2	10.0	18.5	14.3
3	30.0	29.6	29.8
4	10.0	25.9	18.0
5	20.0	14.8	17.4

YEAR LEVEL 10			
	M	F	TOTAL
3	0.0	10.0	5.0
4	0.0	0.0	0.0
5	11.4	10.0	10.7
RARELY	25.7	55.0	40.4
NEVER	62.9	25.0	43.9
BUS	17.1	15.0	16.1
CAR	20.0	20.0	20.0
BIKE	54.3	50.0	52.1
OTHER	0.0	0.0	0.0
YES	14.3	20.0	17.1
SOMETIMES	28.6	50.0	39.3
NO	51.4	30.0	40.7
1	54.3	50.0	52.1
2	20.0	30.0	25.0
3	11.4	20.0	15.7
4	5.7	0.0	2.9
5	8.6	0.0	4.3
1	20.0	0.0	10.0
2	22.9	15.0	18.9
3	22.9	60.0	41.4
4	20.0	25.0	22.5
5	14.3	0.0	7.1

YEAR LEVEL 11			
	M	F	TOTAL
3	0.0	0.0	0.0
4	0.0	0.0	0.0
5	6.7	28.6	17.6
RARELY	40.0	14.3	27.1
NEVER	53.3	57.1	55.2
BUS	26.7	14.3	20.5
CAR	26.7	50.0	38.3
BIKE	53.3	42.9	48.1
OTHER	6.7	0.0	3.3
YES	6.7	21.4	14.0
SOMETIMES	20.0	50.0	35.0
NO	73.3	28.6	51.0
1	40.0	64.3	52.1
2	33.3	14.3	23.8
3	20.0	14.3	17.1
4	6.7	7.1	6.9
5	0.0	0.0	0.0
1	26.7	21.4	24.0
2	26.7	21.4	24.0
3	33.3	35.7	34.5
4	13.3	21.4	17.4
5	0.0	0.0	0.0

YEAR LEVEL 12			
	M	F	TOTAL
3	0.0	3.7	1.9
4	0.0	0.0	0.0
5	8.7	18.5	13.6
RARELY	34.8	40.7	37.8
NEVER	56.5	37.0	46.8
BUS	8.7	22.2	15.5
CAR	30.4	33.3	31.9
BIKE	52.2	55.6	53.9
OTHER	0.0	0.0	0.0
YES	13.0	40.7	26.9
SOMETIMES	30.4	22.2	26.3
NO	56.5	37.0	46.8
1	52.2	63.0	57.6
2	26.1	14.8	20.5
3	4.3	18.5	11.4
4	4.3	3.7	4.0
5	13.0	0.0	6.5
1	26.1	14.8	20.5
2	17.4	11.1	14.3
3	30.4	48.1	39.3
4	13.0	14.8	13.9
5	13.0	11.1	12.1

CROSSING: William-East

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	17	24	70.8	14	18	77.8
8.00 - 9.00	51	67	76.1	33	45	73.3
9.00 - 10.00	56	75	74.7	48	61	78.7
10.00 - 11.00	32	53	60.4	39	48	81.3
11.00 - 12.00	34	48	70.8	42	51	82.4
12.00 - 1.00	43	65	66.2	37	54	68.5
1.00 - 2.00	35	57	61.4	42	61	68.9
2.00 - 3.00	32	42	76.2	24	37	64.9
3.00 - 4.00	27	37	73.0	28	39	71.8
4.00 - 5.00	31	40	77.5	35	44	79.5
5.00 - 6.00	43	69	62.3	43	57	75.4
6.00 - 7.00	14	21	66.7	10	17	58.8

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

CROSSING: Fitzroy-Quay

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	20	30	66.7	22	32	68.8
8.00 - 9.00	29	44	65.9	39	53	73.6
9.00 - 10.00	23	29	79.3	22	31	71.0
10.00 - 11.00	15	20	75.0	12	17	70.6
11.00 - 12.00	15	31	48.4	18	23	78.3
12.00 - 1.00	17	31	54.8	27	34	79.4
1.00 - 2.00	20	26	76.9	16	33	48.5
2.00 - 3.00	13	23	56.5	13	19	68.4
3.00 - 4.00	15	20	75.0	21	28	75.0
4.00 - 5.00	29	44	65.9	32	44	72.7
5.00 - 6.00	31	43	72.1	28	37	75.7
6.00 - 7.00	6	11	54.5	7	9	77.8

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

CROSSING: Denham-Quay

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	11	16	68.8	10	14	71.4
8.00 - 9.00	27	34	79.4	23	39	59.0
9.00 - 10.00	22	29	75.9	17	27	63.0
10.00 - 11.00	12	18	66.7	13	18	72.2
11.00 - 12.00	17	23	73.9	19	27	70.4
12.00 - 1.00	22	30	73.3	17	22	77.3
1.00 - 2.00	21	30	70.0	10	13	76.9
2.00 - 3.00	12	17	70.6	5	9	55.6
3.00 - 4.00	11	18	61.1	9	14	64.3
4.00 - 5.00	32	43	74.4	27	33	81.8
5.00 - 6.00	18	24	75.0	14	19	73.7
6.00 - 7.00	6	8	75.0	4	7	57.1

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

CROSSING: Fitzroy-East

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	21	26	80.8	20	21	95.2
8.00 - 9.00	39	44	88.6	28	32	87.5
9.00 - 10.00	66	72	91.7	50	57	87.7
10.00 - 11.00	43	52	82.7	31	38	81.6
11.00 - 12.00	38	42	90.5	24	26	92.3
12.00 - 1.00	36	40	90.0	39	42	92.9
1.00 - 2.00	22	28	78.6	27	31	87.1
2.00 - 3.00	17	20	85.0	15	19	78.9
3.00 - 4.00	21	26	80.8	21	27	77.8
4.00 - 5.00	35	41	85.4	26	33	78.8
5.00 - 6.00	30	38	78.9	33	37	89.2
6.00 - 7.00	7	16	43.8	5	9	55.6

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

CROSSING: Fitzroy-Bolsover

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	10	12	83.3	7	8	87.5
8.00 - 9.00	23	29	79.3	17	19	89.5
9.00 - 10.00	31	36	86.1	22	27	81.5
10.00 - 11.00	25	29	86.2	48	53	90.6
11.00 - 12.00	31	33	93.9	49	51	96.1
12.00 - 1.00	27	30	90.0	26	29	89.7
1.00 - 2.00	28	32	87.5	27	32	84.4
2.00 - 3.00	33	36	91.7	17	23	73.9
3.00 - 4.00	28	33	84.8	16	21	76.2
4.00 - 5.00	25	38	65.8	25	34	73.5
5.00 - 6.00	25	33	75.8	19	27	70.4
6.00 - 7.00	6	11	54.5	7	9	77.8

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

CROSSING: Bolsover-Denham

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	15	21	71.4	21	28	75.0
8.00 - 9.00	35	45	77.8	38	41	92.7
9.00 - 10.00	42	59	71.2	47	54	87.0
10.00 - 11.00	37	42	88.1	33	37	89.2
11.00 - 12.00	42	51	82.4	23	28	82.1
12.00 - 1.00	33	41	80.5	26	31	83.9
1.00 - 2.00	36	43	83.7	34	41	82.9
2.00 - 3.00	20	25	80.0	18	27	66.7
3.00 - 4.00	25	29	86.2	28	38	73.7
4.00 - 5.00	31	38	81.6	39	44	88.6
5.00 - 6.00	35	45	77.8	37	41	90.2
6.00 - 7.00	7	14	50.0	7	11	63.6

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

CROSSING: Musgrave-Elphinstone

TIME	MEN			WOMEN		
	1	2	BI	1	2	BI
7.00 - 8.00	11	14	78.6	8	10	80.0
8.00 - 9.00	22	25	88.0	19	22	86.4
9.00 - 10.00	32	37	86.5	31	36	86.1
10.00 - 11.00	37	43	86.0	41	46	89.1
11.00 - 12.00	37	40	92.5	30	33	90.9
12.00 - 1.00	41	47	87.2	22	24	91.7
1.00 - 2.00	25	29	86.2	21	27	77.8
2.00 - 3.00	14	21	66.7	27	31	87.1
3.00 - 4.00	20	28	71.4	19	23	82.6
4.00 - 5.00	13	17	76.5	15	18	83.3
5.00 - 6.00	6	11	54.5	8	10	80.0
6.00 - 7.00	3	4	75.0	8	9	88.9

1 = The number of pedestrians using the crossing.
2 = The number of pedestrians using, and crossing within 20 yards of, the crossing.
BI = Pedestrian Behaviour Index.

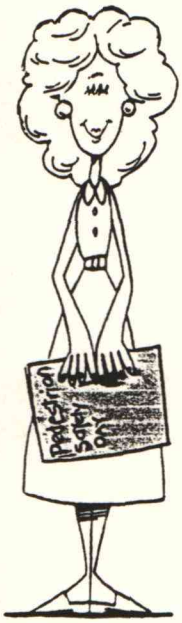
APPENDIX C

Attached is the schedule used by the Road and Traffic Authority, NSW for the schools'"Pedestrian Safety Day"

PEDESTRIAN SAFETY DAY

Teacher's Notes

Years 3-6



Each year in NSW more than 1,000 children are killed or injured on our roads as pedestrians.

A public pedestrian safety campaign for the Sydney metropolitan area is planned to take place during June/July 1990. Schools are asked to support the pedestrian safety campaign by organising their own "Pedestrian Safety Day" during the last week of Term 2.

Use this folder and the STREET SENSE road safety program which is in your school, to plan your "Pedestrian Safety Day". This folder contains "Teacher's Notes" with suggested teaching and learning activities, "Worksheets" for you to photocopy and use with your students, a "Take Home Note" to be photocopied and sent to parents, and a "Pedestrian Safety Sticker" for each student in your class.

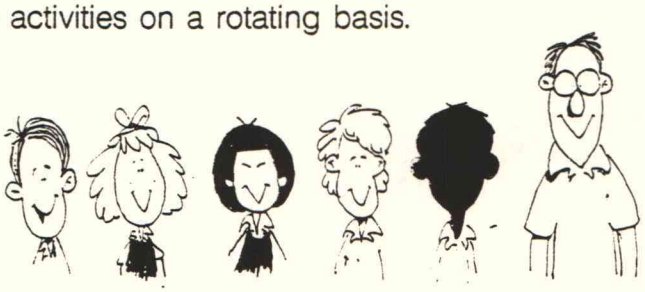
This is a suggested format for your "Pedestrian Safety Day". Each section is optional.

1 Whole Group Activity

Spend 40 minutes or so with the whole group. Watch the Level 2 STREET SENSE video and discuss important points, particularly ones that relate to the road safety environment around your own school. Have a guest speaker. Try the Regional Road Safety Education Consultant or the Police Department.

2 Group Activities (Approx. 20 minutes each)

Find the STREET SENSE kit to use these group activities on a rotating basis. Staff may rotate with the children or remain on the same group. Delete or add further activities according to the number of staff you have available. Refer to the STREET SENSE program for further activities.



a) Driveways and Car Parks

Use the school car park to demonstrate which areas are shared by pedestrians and vehicles.

Refer to STREET SENSE Level 2, Unit 4, page 19.

b) Crossing in the Rain

How does the traffic environment change when it rains?

Refer to STREET SENSE Level 2, Unit 4, page 39.

c) Stopping Distances

A practical playground activity to demonstrate reaction times.

Refer to STREET SENSE Level 2, Unit 4, page 51.

'STREET SENSE' a road safety education program developed and produced by the ROADS AND TRAFFIC AUTHORITY

RTA

d) Using The Footpath (rules and laws)

Discussion and role play.

Refer to STREET SENSE Level 2, Unit 5, page 45.

e) Singing 'To Stop the Cars'

Refer to STREET SENSE Level 2, Unit 5, page 22.

f) Worksheet Activities

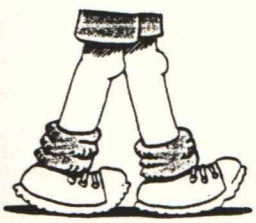
Use the included Worksheets to consolidate pedestrian safety concepts.

3 The Primary Shuffle!!!

Reinforce the concepts learnt in the morning with the Primary Shuffle!!!

Using as many local pedestrian facilities as possible:

* walk to a nearby park or recreational area for lunch. Alternatively set up a pedestrian track in the playground.
* make sure local pedestrian features are discussed and demonstrated to the children.
* involve parents and the local community.
* play games if you wish after lunch and then walk back along a different route.



Many schools have decided to use this idea as the basis for their **school walkathon** and children could be sponsored at a rate per half a kilometre. In this way schools are able to meet important educational needs of their children whilst raising funds for school resources. **It's also a great way to finish the term!!!**

4 Design a Poster

Ask the children you are working with to design a pedestrian safety poster with a caption. Choose the best five and send them to 'Design a Poster', Roads and Traffic Authority, P.O. Box 51 Milsons Point, 2061. Make sure your school's name, with the name of a contact person, is included.

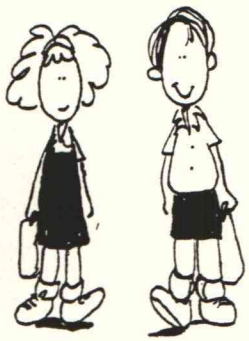
The best entries will be chosen to be in next year's STREET SENSE Calendar and will also be displayed in shopping centres during the July school holidays as part of the Pedestrian Safety campaign.

PEDESTRIAN SAFETY DAY

Take Home Note

Years 3-6

Dear _____



At school I've been learning about pedestrian safety.

Did you know that each year over 1,000 children are killed or injured as pedestrians?

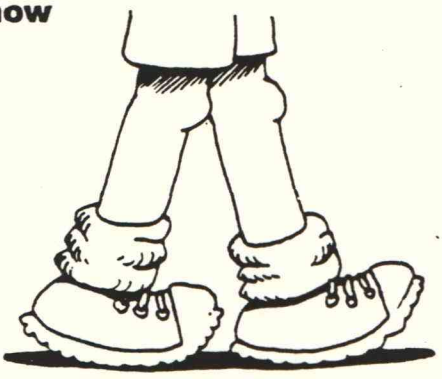
I'm going to draw a plan of my neighbourhood, or the neighbourhood near my school, on the back of this sheet.

I'm going to mark the safe and unsafe places for pedestrians to cross the road in the neighbourhood. If I walk to school I'm also going to mark the safest route for me to travel.

Would you please:

check my pedestrian safety plan and sign it so that I can take it back to school to show my teacher.

Next time we're out walking in the neighbourhood we can test each other to see if **you** are as safe a pedestrian as I am. **You know it's not just kids who have trouble crossing the road safely!**



Signed _____ **(Parent or Carer)**

'STREET SENSE' a road safety education program developed and produced by the **ROADS AND TRAFFIC AUTHORITY**

R T A

U.C.C.G.

CAPRICORNIA INSTITUTE OF ADVANCED EDUCATION

LIBRARY

The undersigned have consulted this report for private study and research, and undertake not to copy it in whole or in part without the author's permission.

Signature Redacted