A REPORT LANGUAGE AND INTERPRETER FOR DATA ANALYSIS

by

L. D. Lubin
Department of Computer Science
University of Manitoba
Winnipeg, Manitoba
May 1974

A Report Language and Interpreter for Data Analysis

A Thesis Presented to
The Faculty of Graduate Studies and Research
The University of Manitoba
Winnipeg, Manitoba

In Partial Fulfilment
of the Requirements for the Degree
Master of Science
in the Department of Computer Science

ACKNOWLEDGEMENTS

I would like to thank Professor P. H. Dirksen for his advice during the writing of this thesis. His constructive criticism was always a help. I would also like to thank my wife, Ruthann, for her help in proof-reading and typing this thesis and Miss S. Elo of the University of Toronto Computer Centre who typed the final version.

Thanks are also due to the computing centres of both the University of Manitoba and the University of Toronto for allowing me full access to their facilities.

TABLE OF CONTENTS

INTRODUCTION

1.1 Nature and Scope of Thesis

The area of computer languages has been and will continue to be a very important area of research in the computer field. We have low-level and high-level languages for almost all types of specialized as well as general purpose applications. We even have languages to help implement other languages. With all of these languages, though, there are still areas where a computing need exists but the solution of a problem is made difficult because of a lack of software facility. The specific area in question here involves data analysis and extraction by a researcher not oriented toward computer analysis.

This thesis describes the design and implementation of a system that allows a researcher with little or no exposure to a computer to use the machine as a tool to help gain insights into the data he is examining. This system was implemented on an IBM System/360 Model 65 computer at the University of Manitoba and an IBM System/370 Model 165 computer at the University of Toronto Computer Centre.

1.2 Organization of Thesis

This thesis is divided into three main parts:

(i) an historical review

(ii) the language definition

and (iii) the language implementation

Readers wishing to write programs using the system will be mainly interested in sections (ii) and (iii) corresponding to chapters 3 and 4, where the language and its implementation is discussed in detail. The user will find a complete program in the appendix, written to show the features of the system.

HISTORICAL REVIEW

2.1 The Problem

The researcher not familiar with computers faces a difficult task when trying to analyze medium to large amounts of data. Numeric computations can be very time consuming and most researchers prefer to expend their efforts on other aspects of a project. With large amounts of data, the researcher may not have time to do as complete an analysis as he needs to get relevant information from his data base. There are three basic choices he can make.

1) He can try to find which of the many languages in existence is best for his requirements. Then, if the language is available on a machine he has access to, he can proceed to learn the language and do his analysis.

The main fault with this choice is the amount of work that must be done by the researcher before he gets any results. This method does not usually give the researcher any preliminary results that would enable him to decide whether or not his data is significant and whether he should spend the time and money for a more detailed study. There is also difficulty in choosing the proper language. Some of the widely used general-purpose languages, FORTRAN or PL/1 for example, will produce the results needed but the researcher will have to learn far

more about the language than he probably needs. If, on the other hand, a more tailored language is found, one that specifically suits his needs, the researcher may run the risk of learning a language that he may never have need of again. Usually, none of these language types is very useful for a preliminary type of analysis.

2) Researchers in the social science fields often want to count the number of records that correspond to specified conditions. This type of analysis may be done using a card sorter. However, if the number of conditions that are to be met is more than a few, the researcher can be wasting a lot of time getting few results. This method is sufficient for some types of data extraction but requires at least one pass, and usually more, over the data for each result required.

3) The researcher's last alternative is to try to reduce his data tc a form where it may be manually tabulated. Data or variables thought to be insignificant could be deleted when they are, in fact, significant. Mechanical errors inherent in manual numerical calculations could lead to faulty inferences. It would be a waste not to take advantage of the powerful tools we have access to.

2.2 Other Solutions

Not very much work has been done to aid the researcher and especially the social science researcher in his efforts to use the computer as a scientific tool. Of what little has been done, five languages have been chosen to indicate what is available for the type of research discussed. These languages each show a different philosophy in dealing with the non-computer-oriented researcher and the social science researcher wishing to do a preliminary analysis. The five languages are:

1) SRG-40 SURVEY REPORT GENERATOR
2) STATISTICAL REPORT GENERATOR
3) The DATA-TEXT Language
4) COBOL
5) APL

2.2.1 SRG-40 SURVEY REPORT LANGUAGE

SRG-40 was written for the IBM-7040 computer at the University of Waterloo. Its objective was to simplify data collection and report writing by computer. Program preparation was to be kept to a minimum without sacrificing efficient use of the computer.

Data input to an SRG-40 program must be in one of two forms, coded or actual. With coded data, a pre-

assigned code is used to indicate an item of data in place of the actual value, for example, 1 for male and 2 for female would be adequate codes for a piece of data indicating a certain sex. Actual data indicates that the actual value of an answer is used to represent itself, for example, the number of people over the age of forty in a town or the total salary of a family. Codes can be one or two digits or characters and actual values can be from one to nine numeric characters.

SRG-40's faults lie with its limitations and restrictions. The language has only two basic commands, accumulate (i.e. sum) numeric fields and count the occurrences of different coded combinations. This leaves the language uncomplicated for the non-programmer but does not offer him a powerful enough utility. A language should be simple and powerful.

The format of instructions leaves a lot to be desired. Almost everything is defined in terms of numeric codes, making the instructions unintelligible to a person not familiar with the language. This also makes the language harder to learn and remember. Sample commands are as follows:

1) C2010400100100200200300300400 4*

2) A645GR09500*

In this form they are totally meaningless.

The first example would classify part of the data being examined into four categories. The number of times these classifications occurred would then be tabulated.

The "C" indicates we have a code variable. The "20" indicates the starting column where the data may be found and the "1" indicates length. "04" tells how many coded groups are defined. The remaining digits define the actual data expected and a numeric equivalent. Each statement is ended by an *.

The second example can be used to sum the values of a specific data field if the data is more than 9500. The "A" defines an actual variable that is to be accumulated. The "64" indicates the data location and "5" indicates field width. "GR" specifies a comparison that is to be performed on the data and if the comparison holds true, the accumulation takes place. In this case if the data is greater than "9500" the data will be added into the total. The only comparisons allowed are GR and LE, greater than and less than or equal.

The two statements could be run separately or combined with other statements to count and accumulate based on a number of fields, i.e. count one field AND another field AND another and possibly accumulate the values in a number of other fields.

The chairman of the Sociology department at the University of Manitoba was asked to give his impressions of the needs and capabilities of social scientists with respect to computer analysis of their research data. He said that social scientists are not accustomed to computer programming although the situation is improving. When asked what should be included in a language for social scientists, he replied that the statistical capability should be limited, no tests of significance but possibly some histogram capability. He reasoned that there were enough statistical packages available and that the researcher needed a pre-statistical capability for counting and tabulating his data.

To this point, the philosophy of SRG-40 agrees, although there is no plotting capability in the language. It is also at this point where the two philosophies diverge. SRG-40 provides no error checking. Program cards must obey format rules and data must be error free. This is contrary to the above belief and the general belief today that languages should be easier to code and provide thorough error checking and good diagnostics. This is especially true for the type of researcher this language is aimed at.

There are a few other faults with SRG-40 that detract from its utility. The format of the report generated by the program is very difficult to read. A

researcher should not have to search for his results when they have been tabulated and are sitting in front of him. Data input to the program must be on cards. Although the majority of social science research data is punched and kept on cards there is no facility available for the times when this is not the case. An even bigger problem is the fact that a user must have one and only one card per record. This means a researcher can only have a maximum of eighty columns of information analyzed at one time.

SRG-40 was developed around 1965 and it should be kept in mind that it was written for a relatively small machine, thus limiting its capabilities.

2.2.2 STATISTICAL REPORT GENERATOR

The second language to be looked at is STATISTICAL REPORT GENERATOR which will be called SRG (as opposed to SRG-40). Although its name is similar to SURVEY REPORT GENERATOR and they were both developed at the University of Waterloo, there is very little else that is similar.

SRG was designed for researchers needing statistical information and reports but who are not familiar with Data Processing concepts. SRG is a total system, not just a language. In fact, the actual program to analyze data is only one-third of the system. It is this part that is most relevant to the discussion and

will be dealt with in greater detail. SRG moved away from the numeric instructions codes of SRG-40 to a more readable English-like format. Instructions are not completely readable but make more sense and are easier to remember. SRG gives users the capability of counting the occurrences of different data combinations as well as summing numeric data items as in SRG-40. The user also has the power to define his own statistical requirements in terms of mathematical expressions. The user has the capability of choosing which data items are to be output in the final report. Sample instructions are as follows:

```
PRINT NAME AGE SEX
COMPUTE AVG = TOTAL/NUMB
COUNT
IF CONDITION: (1) COUNT
```

With the last instruction, a count is made only if the previously defined condition, namely condition number 1, is satisfied. The condition could be SEX.EQ.'F' if a count of females is desired or AGE.GT.40 if a count of people over the age of forty is required. Any instruction except another IF may follow an IF CONDITION. As can be seen, SRG is easier to read and remember than SRG-40.

Like SRG-40, though, SRG has its faults. The language is designed for a terminal system which means

the user must learn the terminal commands and operation as well as the language. Many installations do not support terminals at all, making it impossible to use. In addition to this, the files that are to be analyzed must be set up off-line. This adds another part of the system to what a user must learn before he can get any results. Altogether, this makes for a fairly complex system, more complex than is needed for a beginning programmer.

The instruction set is fairly easy to use and only has one flaw as far as counting occurrences of different combinations of data goes. If a researcher wants to find the number of occurrences of all permutations of classes of data, he must specify each possible combination in an IF statement. This means a great amount of work for this type of tabulation.

2.2.3 DATA-TEXT

Another language that is discussed is DATA-TEXT. This language, developed about 1963 at Harvard University, and since substantially revised (about 1970) for the IBM 360 and CDC 6000 series machines, was designed specifically for data analysis in the social sciences.

The language is very powerful and imposing because of its size. The documentation for the language is so

extensive and voluminous that a beginning programmer has a tendency to feel swamped before he starts. The language offers the researcher almost any and every statistical analysis available. Some of these are listed as follows:

1) Simple statistics - mean, standard deviation, variance, count
2) Frequency Distributions
3) Scatterplots
4) Correlations
5) Cross-tabulations and Cross-statistics
6) Factor Analysis and Rotations
7) T-Tests and F-Tests
8) Multiple Regression
9) Analysis of Variance

Data to be input to the language may reside on cards, disk or tape. Variables, representing units of data in a record of a file, are defined in a simple, readable manner. The user gives each data item a distinct name, specifies its location in a record, describes the data item in an English-like manner and describes its codes if it is a coded piece of data.

Actual statistical routines are specified in COMPUTE statements as follows:

1) *COMPUTE STATISTICS(AGE,INCOME)
2) *COMPUTE T-TESTS(ABILITY)

3) *COMPUTE CROSSTABS(RACE,ITEM BY SEX,CLASS)

4) *COMPUTE REGRESSION(ABILITY ON AGE,SEX,INCOME), GROUP BY CLASS

As can be seen, the statistic required is followed by a list of variables to be tabulated along with other needed information. Instructions are readable and fairly easy to remember.

The main fault with DATA-TEXT is its size. The researcher wishing to do a pre-statistical analysis does not need all that the language has to offer. There is so much available to the user it is difficult to find just what is needed. DATA-TEXT programs can be costly in terms of computer space required. On an IBM machine, DATA-TEXT programs need approximately 200K bytes before they can run.

DATA-TEXT has been well planned and thoughtfully implemented but it is more suited to the experienced programmer, not the researcher unfamiliar with programming or existing statistical packages.

2.2.4 COBOL

COBOL stands for Common Business Oriented Language. Its chief functions are the maintaining and handling of large complex data files, the generation of management reports and numerous accounting applications. It is disliked by many scientific and mathematical programmers

and owes much of its success to the support it received from the United States government, the world's largest data processing user.

Programs give the appearance of English and are primarily self-documenting with each program divided into four major sections or divisions. Two of these divisions, IDENTIFICATION and ENVIRONMENT, are used for documentation purposes. The DATA DIVISION is used to describe files and variables in a detailed way. The final section, the PROCEDURE DIVISION, is used to describe the calculations that are to be performed.

COBOL is one of the most widely-used languages in existence today. It is relatively easy to use and provides a variety of facilities for the business programmer. When used in a non-business-oriented application, the language has a few drawbacks. Its self-documenting nature requires the programmer to provide information that is not always required.

Statements are often of a fixed format that must be started within a specific range of columns and must end in a certain fashion. For example, the IDENTIFICATION DIVISION must include a PROGRAM-ID starting between columns 8 and 11 followed by a period and a space. The program name then follows enclosed in single quotation marks or apostrophies. This is then also followed by a period and a blank. The language offers such a wide

variety of data types that a user must be aware of not only the type of data he has but in many cases its internal machine representation as well. Many facilities are available to output these data types in a report format. COBOL is hardly every used for mathematical or statistical work. The language has the capability of doing this type of work even though it is, at times, awkward.

2.2.5 APL

APL is "A Programming Language" and is the last to be discussed. The language was implemented by IBM for IBM systems back in the 1960's and is only now being made available by other companies.

Implementation has been as a remote terminal time-sharing interpretive system and thus its use is further restricted to those systems that provide terminal/tele-processing services.

The last year or two has seen a drive, partly spearheaded by Harlan Mills of IBM Corporation, to convert programmers and programming language design to a philosophy called structured programming. Using the proper language and techniques, Mills and others have shown that structured programming dramatically increases programmer productivity as well as the reliability of his work.

APL is the opposite of almost everything structured programming stands for and can be a very difficult language to use or maintain. Variables are defined in much the same way as other languages, that is a combination of letters and/or numbers, starting with a letter. The standard operators are used, +-/x as well as a number of special operators and Greek letters for vector and matrix calculations. When they are all used together it can be very difficult to decipher a program. The following example will help clarify this point. Given a vector of elements x, the line of code shown is sufficient for calculating the average of the elements of the vector.

$$(+/\times)\div\rho\times$$

$\rho\times$ gives the number of elements in the vector x

$+/\times$ sums the elements in the vector and

$(+/\times)\div\rho\times$ does the final division to calculate the mean

As problems become more complex, their functional descriptions also become more difficult to understand and read. For the person who does not wish to become bogged down in complex programming problems, APL is not a solution.

The researcher who has large files of data is not given satisfactory service either. At present, there is only one company in Canada, Toronto-based I.P. Sharp, that offers any file-handling capability to speak of. IBM has announced a file handling service for their new

virtual system version of APL but it won't work for almost all of the existing systms that are running today.

In summary, a person wishing to do a short statistical summary either has to learn too much about the language or is not offered the necessary facilities if he wants to use APL.

DARE - A DATA REPORT LANGUAGE

3.1 Philoscphy

It was felt that the computer languages that existed were not sufficient for solving the type of problem discussed. Throughout the development of DARE (A DAta REport Language), every design concept tried to encompass the following principles.

(1) Simplicity

DARE was designed to be used as a research tool, mainly by social scientists more interested in their data than in "programming". The language has default values for many options and instructions thus helping to cut down on the coding needed. The grammatical structure, or syntax, follows a logical pattern helping to guide the researcher toward his goal. Unlike certain languages such as COBOL, DARE does not require a large amount of coding to do a little work. With DARE, a researcher can do detailed analyses with relatively simple instructions.

(2) Utility

DARE was not designed to replace statistical packages. It was felt that there were more than enough to handle almost any statistical problem a researcher would encounter. Many researchers wanted some type of

pre-statistical analysis, file editing capability and counting facility. DARE gives a user all of these within a variety of formats and options. DARE was specifically designed for the users who previously spent hours at a card sorter, sorting their data into different classifications or the users who wanted to know if their data had certain preliminary characteristics before they did a lengthy statistical analysis.

(3) Flexibility

The non-computer-oriented researcher was kept in mind constantly when DARE was being developed. Instructions required only a minimum of information to produce a good deal of analysis. This is fine for many researchers but was still inadequate for those who wanted an improved result. With this in mind, DARE was designed with "options", sections of code that were not necessary for the basic running of the program, but, when used, enabled the user to have a more comprehensive or more readable result.

(4) Generality

DARE was originally designed to be machine independent. None of the language's instructions or data types have any bearing or relation to a specific machine or implementation style. Only minor modifications are required to implement the existing version on a wide range of machines.

It was with these four principles that DARE was created. Each served as a guide and played an integral part in the development of the language.

3.2 Language Description

Programs written in the Data Report Language (DARE) have two major sections:

(1) File Definition

and (2) Tabulation and Report Writing

3.2.1 File Definition

Every file, or set of data that is to be used, must be defined in the File Definition section. To distinguish one file from another, the files are named in the following manner.

FILENAME (FILEnn)

nn is a two-digit number from 01 to 20. Numbers 01-05 have been reserved for input files (files which will be read in) and numbers 06-20 have been reserved for output files (files which are created by the program). Since most researchers have their data on cards and wish their results on the printer, FILE05 and FILE06 have been reserved for card input and printer output respectively. Users whose data is stored on magnetic tape, disks, or other devices or who wish to create new files on these devices, define their files in the exact same way as card or printer files.

Up to now, the user has named the file and, in so doing, specified whether it is input or output. He must also describe each field in the file that he wishes to use. Every field must have associated with it a distinct variable name along with other essential information such as its location and data type. This constitutes the minimum the user must declare. In addition to this, each variable can have an associated "variable description" for aid in documenting the program and providing a more aesthetic report. Some variable names can also have a "coded description" and this will be discussed at greater length shortly. The following is the format for defining a variable:

VARIABLE NAME = LOCATION, TYPE, 'VARIABLE DESCRIPTION', (CODED DESCRIPTION)

Anything underlined is optional.

The different sections are now described.

(1) Variable Name

The variable name is the name a user associates with one piece of data or one field in his file. It may be a combination of one to six letters and numbers, but must start with a letter. (Some valid variable names are: ADDR, SEX, PROV1, PROV2, P109X, and P74A3.) The following are invalid variable names and the reasons why they are in error are given.

ADDRESS - too long, more than six characters

CODE#1 - contains the invalid character #

123HI - starts with a numeric character

It is suggested that users try to choose meaningful variable names, ones that have some connection with the data they represent. This makes debugging and documentation of the program much easier for the user.

(2) Location

A field's location is separated from the variable name by an equal sign. This indicates both the end of the variable name and the beginning of the column specification.

A variable's location in the record is indicated by giving the column number that represents its position in the file. An item that covers more than one column is specified by giving the first and last column numbers separated by a hyphen. If an item occupied positions 21 through 25 in a file, these would be indicated by 21-25. A single column would be represented by a single number. The column location is enclosed in parentheses () and is preceded by the word COLUMN or COLUMNS. The short forms COL or COLS may also be used. The following four examples are all equivalent as far as the definition of the language is concerned.

NAME = COLUMNS (10-25)

NAME = COLUMN (10-25)

NAME = COIS (10-25)

NAME = COL (10-25)

(3) Type

The type specification is used to indicate the fundamental characteristics of the data represented by the variable name. In keeping with its objective of simplicity, DARE only allows three data types, alphameric, numeric and coded.

Alphameric variables represent data that can be composed of alphabetic, numeric or special characters. Its main use is for input and output only. Data items that might normally be coded as alphameric (sometimes called alphabetic) are names, addresses, etc.

Numeric variables represent data items that are numbers and nothing else. Generally, the researcher wants some arithmetic operation performed upon these items. Examples of numeric fields could be age, salary, number of children, etc.

The final type specification is coded. This is a special specification that gives DARE most of its power. Often a researcher groups his data into classes according to some criterion. He can then represent each class by its own specific code. For example, a sociologist may be doing a study on the differences in

the cost of living in different regions of Canada. He could subdivide the country into the following regions:

1) Maritimes,
2) Quebec,
3) Ontario,
4) Prairies,
5) British Columbia, and
6) North.

The user could then specify the region that a record applied to by specifying the number that was associated with its region or in this case, even the first letter of the region's name would suffice as a code. M could stand for the Maritimes, Q for Quebec, etc. Codes can be either alphabetic or numeric and are not restricted to being one letter or digit in size.

The actual type specification is indicated by using the words ALPHAMERIC, NUMERIC or CODED after the column specification. A few short forms and alternatives have been included to help make the programming and preparation less tedious. ALPHABETIC may be used as an equivalent form of ALPHAMERIC and both may be indicated by the letter A. NUMERIC and CODED may be represented by N and C respectively. If the type specification is omitted completely, the variable is assumed to be CODED.

The type specification can occur immediately following the column specification or they may be separated by either a comma or blanks. It is advisable to use some type of separation for readability.

The following examples show correct ways of defining variables.

```
1)   NAME =  COLS(1-15), ALPHABETIC
2)   ADDR =  COLS(16-30)A
3)   CITY =  COLS(31-45) ALPHAMERIC
4)   SALARY =  COLUMNS(46-50), NUMERIC
5)   AGE  =  COL(51-52)N
6)   SEX  =  COL(55) CODED
7)   PROV =  COL(56)      ...(CODED assumed with no
                                specification)
```

(4) Variable Description

In addition to the above information a user may add a variable description field. This is a character string enclosed in single quotes (') that helps to describe the variable name. This description is used to aid in the readability of the output results. If the description contains a quotation mark, it should be entered as two single quotation marks. The following could be used as descriptions for the above variable names:

```
1)   'NAME OF FAMILY HEAD'
2)   'STREET ADDRESS'
3)   'CITY OF RESIDENCE'
4)   'TOTAL INCOME OF FAMILY'
5)   'FAMILY HEAD''S AGE'
6)   'SEX OF FAMILY HEAD'
7)   'PROVINCE OF RESIDENCE ON DECEMBER 31'
```

These descriptions can immediately follow the type of specification but it is again suggested that the user use either a comma or blanks to separate it from the type specification.

(5) Coded Description

The final specification is the coded description and can only be used with coded variables. It is a means of specifying exactly what each possible code stands for in a coded field. The description should take the form:

(CODE/DESCRIPTION,CODE/DESCRIPTION...)

with one CODE/DESCRIPTION for every possible code that is used. An example showing the definition of a variable representing a regional code will help make this clearer.

REGION=COLUMN(36) (0/MARITIMES,1/QUEBEC,2/ONTARIO,3/WEST)

Note that the definition here does not need a type specification as CODED is assumed. Also, there is no variable description but this is not necessary either. The example shows the column specification and coded description separated by a blank. As with the other specifications, this could have been a comma or no space need have been left at all. Each possible code has been listed as a CODE/DESCRIPTION group. This is the best way to code the group even though only the CODE section is necessary. The /DESCRIPTION part helps to document the variable and makes the output more readable. Some codes are self-documenting and require no description or explanation. For example, the following provincial code will demonstrate this:

```
PROV=COL(1-5),CODE,'PROVINCIAL CODE',(PEI.,NFLD.,NS.,
                                      NB.,QUE.,ONT.,MAN.,
                                      SASK.,ALTA.,B.C.)
```

At this point it might be helpful to describe an entire input file. Every file definition and variable definition must start on a new line. Since card input is the most widely used form of program input, each file definition and variable definition will start on a new card. They may start in any column and continue over to the next if necessary. Only card columns 1 to 71 should be used for program statements. If an instruction or definition is too long to fit in the 71 columns, punch any character in column 72 and continue on the next card as if it were an extension of the one being prepared.

Consider a file of student histories containing the student's name, city of residence, age group, sessional grade and cumulative grade. The data is assumed to be on cards. This file could then be defined as follows:

```
FILENAME(FILE05)
     NAME=COLS(1-15),A,'STUDENT'S NAME'
     CITY=COLS(16-25),A,'CITY OF RESIDENCE'
     AGE=COL(26),'AGE GROUP',(0/UNDER17,1/18-20,2/OVER 20)
     GRADE1=COL(27),A,'SESSIONAL GRADE'
     GRADE2=COL(28),A,'CUMULATIVE GRADE'
```

Note that the variable definitions are indented from the file definition. This has no effect upon the program and is only used to aid readability. Any file can now be defined using the specifications described above. If a researcher is only in need of a subset of the data he

has in a record, he only needs to define the variables that apply to the information being used. Once the files are defined, the researcher can then move on to the next section, the tabulation section, where he describes what to do with his data.

3.2.2 Tabulation Section

The tabulation or processing section of a program is the section that specifies what work is to be done; what operations are to be performed.

Eight different operations are available to the user to accomplish his evaluation and preliminary statistical analysis of his data.

The instructions are:

1) READ
2) WRITE
3) MOVE (TRANSFER data or ASSIGN)
4) CALCULATE
5) CLASSIFY
6) ORDER
7) PLOT
8) IF

3.2.2.1 READ, WRITE, DATA TRANSFER

The READ and WRITE instructions will be discussed together because of their great similarity. The format of these instructions is:

instruction (filename)

for example, READ(FILE05) or

WRITE(FILE06)

These would cause the information currently stored in the variables associated with FILE05 and FILE06, as specified in a preceding FILE section, to be input and output respectively. File numbers are tested to ensure a READ only accesses an input file and WRITE only accesses an output file. Users are also notified about errors in format and what is expected. The statement

READ) FILE05)

would generate the error message

*** ERROR *** EXPECTING (BEFORE FILENAME BUT NOT FOUND

Every attempt has been made to make diagnostics as meaningful and helpful as possible. Other error or warning messages associated with input and output specifications are:

1) IMPROPER CHARACTER FOLLOWING FILENAME

2) IMPROPER FILE SPECIFICATION FOLLOWING (

3) INCORRECT FILE NUMBER. MUST BE 2 NUMERIC DIGITS

4) FILE NUMBER OUTSIDE RANGE 1-20

5) NO FILE HAS BEEN SPECIFIED OR FILE DEFINITION MISSING

Because entire files are input and output by a single instruction and file structures are defined with unique variables, there must be a way of transferring data from one variable to another. A MOVE or assignment statement is used for this transfer and has the following structure.

variable1=variable2

When this statement is executed the value of variable2 is assigned to variable1. Both variables must be of the same type. For coded or alphanumeric data, truncation or padding will take place if necessary at execution.

Extensive error checking is carried out when this statement is translated yielding the following error or warning messages.

1) A VARIABLE IN THE ABOVE STATEMENT DID NOT APPEAR IN THE FILE SECTION

2) MIXED MODE IN ASSIGNMENT STATEMENT. VARIABLES DO NOT HAVE THE SAME TYPE

3) TRUNCATION WILL OCCUR AT EXECUTION TIME. CAUSE: FIELD SIZE OF VARIABLES IS NOT THE SAME

4) PADDING WILL TAKE PLACE AT EXECUTION TIME. CAUSE: FIELD SIZE OF VARIABLES IS NOT THE SAME

3.2.2.2 CALCULATE

The READ, WRITE and MOVE statements are very necessary to any computer language but they are not the focal point of this language. The CALCULATE statement offers the user the basic functions of summing or totalling fields, calculating averages and standard deviations. The statement takes the form:

CALCULATE operation (OF) VARIABLE1 (BY VARIABLE2)

Parts of the instruction enclosed by parentheses( ) are optional. Choices for operation are:

1) SUM or TOTAL

2) AVERAGE

3) STANDARD DEVIATION

VARIABLE1 must be a numeric variable and VARIABLE2 must be CODED if used.

Sample statements are as follows:

1) CALCULATE TOTAL PEOPLE

2) CALCULATE SUM OF AGES

3) CALCULATE AVERAGE AGE

4) CALCULATE STANDARD DEVIATION OF ERRORS

5) CALCULATE AVERAGE AGE BY CITY

6) CALCULATE TOTAL WRKDAY BY MONTH

In statements without the BY specification, the numeric operation yields one result, the one that was requested. In the statements with the BY specification, one numeric result is produced for each of the codes of the coded variable as it was defined in the file section. In the fifth example above, if ten cities had been defined in the file section for the variable CITY, ten averages would be calculated for this statement, each average pertaining to a single city.

Sample diagnostics that can be generated during translation of this statement are:

1) EXPECTING ARITHMETIC FUNCTION AFTER "CALCULATE" BUT IT WAS NOT FOUND

2) EXPECTING VARIABLE NAME OF NUMERIC ITEM AFTER OPERATION BUT NOT FOUND

3) EXPECTING NUMERIC VARIABLE BUT CODED OR ALPHABETIC FOUND

4) EXPECTING "BY" SPECIFICATION OR END OF CARD BUT NON-ALPHABETIC CHARACTER FOUND

5) UNDECODEABLE STATEMENT

6) CODED VARIABLE EXPECTED AFTER "BY" BUT NUMERIC OR ALPHABETIC FOUND

7) EXPECTING CODE AFTER "BY" SPECIFICATION: IT HAS BEEN IGNORED.

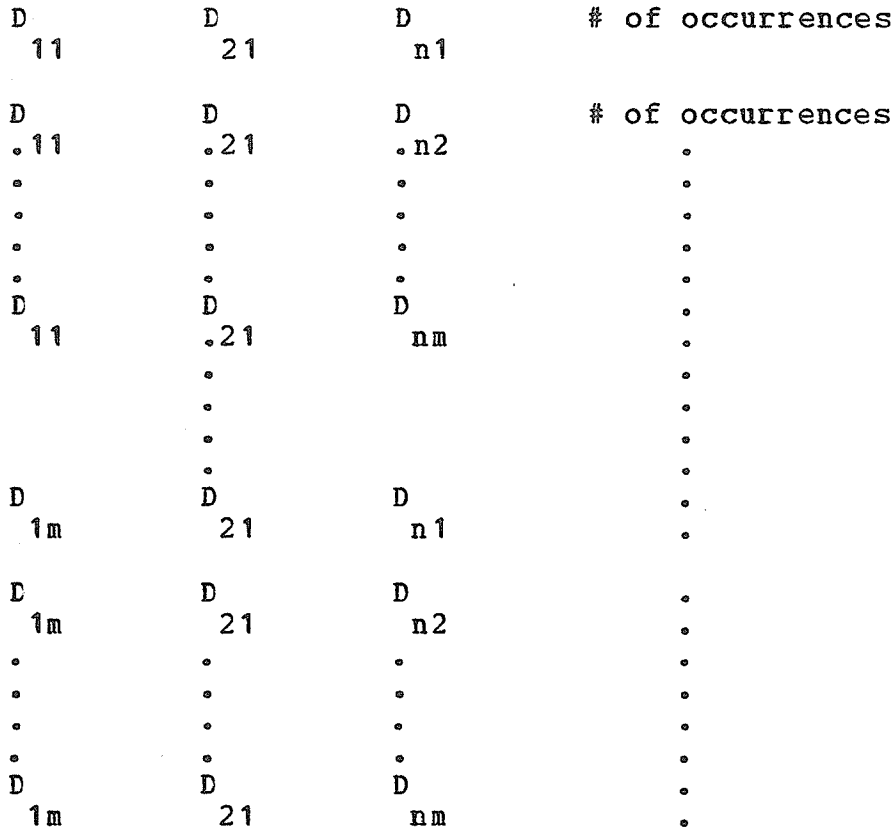
3.2.2.3 CLASSIFY, ORDER, PLOT

Up to this point the programmer has been able to input and output his data and do the elementary arithmetic functions that make up part of the objectives of this language. The remaining needs of the researcher within the objectives specified could be satisfied by some type of counting and/or histogram plot facility.

The main method available for counting occurrences of specific data is the CLASSIFY statement. A user can determine the number of times a specific combination of up to twelve variables occurred. Each of the up to twelve variables must be of the CODED type and the instruction yields a matrix-like result. The CLASSIFY statement takes the form

CLASSIFY(variable1,variable2,...variable12)

where all variables are of the CODED type and at least one is listed between the brackets. If D_{ij} represents the jth code of the ith variable of the classification, the results will be listed in the following form:



Every possible combination of all of the codes of all of the variables in the CLASSIFY statement is listed. This statement requires the greatest amount of internal storage of any of the instructions. It was felt that more use would be made of the information that was generated. Often a researcher would like to be able to regroup the variables specified in the CLASSIFY statement so that the combinations that result are more functional for his individual requirements. To this end, another statement was included. The ORDER statement has the exact same format as the CLASSIFY statement and actually refers to the CLASSIFY statement that precedes it in the program. The ORDER statement contains the same

variables as the previous CLASSIFY but in a different order. An example of these two types of statement is as follows:

```
CLASSIFY(A,B,C)

ORDER(A,C,B)

ORDER(B,A,C)
```

The ORDER statement functions in the exact same manner as the CLASSIFY except in one very important respect. The results for the ORDER statement are produced at "PRINT" time and no major storage or execution time is used by the statement. All data that is needed is accumulated by the CLASSIFY statement and when the program is complete, the ORDER statement just re-orders it.

Checking is carried out to insure that all variables used by both statements are CODED and the syntax is correct. Extra checking is carried out in the ORDER statement to insure that the variables listed agree in name and number with the previous CLASSIFY.

Possible error or warning messages issued are:

1) EXPECTING (AFTER CLASSIFY BUT WAS NOT FOUND
2) MISSING VARIABLE NAME. EXPECTED CODED VARIABLE BUT NOT FOUND
3) TOO MANY VARIABLES IN CLASSIFY INSTRUCTION,

MAX. IS TWELVE

4) VARIABLE USED IN CLASSIFY INSTRUCTION WAS NOT CODED TYPE

5) "," OR ")" EXPECTED AFTER VARIABLE NAME BUT NOT FOUND

6) NO CLASSIFY PRECEDED THE FIRST ORDER STATEMENT

7) EXPECTING (AFTER ORDER BUT NOT FOUND

8) VARIABLE IN ORDER WAS NOT IN PREVIOUS CLASSIFY

9) NUMBER OF VARIABLES IN ORDER STATEMENT IS NOT THE SAME AS LAST CLASSIFY

The final statement to be described is the PLOT statement. In both of its two forms, the PLOT statement gives the researcher a histogram plot of either CODED or NUMERIC data. The statement takes the following forms:

1) PLOT BY variable1 where variable1 is a coded variable

and 2) PLOT variable2 GROUPS=#1,RANGE=#2-#3
where variable2 is a numeric variable.

If the PLOT statement is used with BY and a coded variable, a histogram plot is generated with one bar of the graph representing the occurrences of each code of the coded variable as it was described in the file section.

If the PLOT statement is used with a numeric variable the user must specify a range or upper and

lower bound for the values which he wants included as the x-axis of his plot. If only one value is specified it is assumed to be the upper bound with the lower defaulting at zero. The user must also specify the number of groups (corresponding to bars on his graph) into which the range is to be broken. The y-axis of the graph (i.e. the height of the bars) represents the number of times the value of the variable specified fell into the group plotted.

As with all of the other statements, extensive error checking is carried on with the PLOT statement as well. Error messages generated are as follows:

1) ILLEGAL CHARACTER SEQUENCE AFTER "PLOT"
2) ILLEGAL CHARACTER SEQUENCE AFTER "BY"
3) ALPHABETIC VARIABLE FOUND IN PLOT STMT
4) EXPECTING "GROUP" OR "RANGE" SPECIFICATION BUT NOT FOUND
5) GROUP SPECIFICATION ALREADY ENCOUNTERED
6) EXPECTING "=" AFTER GROUP BUT NOT FOUND
7) EXPECTING NUMERIC VALUE AFTER GROUPS
8) NO RANGE SPECIFICATION WITH NUMERIC VARIABLE
9) NO GROUP SPECIFICATION WITH NUMERIC VARIABLE
10) EXPECTING NUMERIC RANGE SPECIFICATION BUT NOT FOUND
11) ILLEGAL SEQUENCE FOLLOWING RANGE SPECIFICATION
12) TOO MANY NUMERIC BOUNDS FOR RANGE

3.2.2.4 IF

At this point the user has all of the basic commands needed to do an introductory statistical analysis of his data. He can READ, WRITE, CLASSIFY, PLOT, and calculate results in a relatively efficient and easy-to-use manner. It was felt, though, that one additional facility should be added to the language to give it added versatility from the user's point of view. There would always be a time when he would like to apply the existing commands to a specific subset of his data. The best way to do this was to provide a FORTRAN-like IF statement that could be used with all of the commands listed. The IF statement takes the form

IF(logical expression) command

where command is one of the previously described instructions and the logical expression is a combination of operators and operands that when evaluated yield a result of "true" or "false". Operands can be variable names, alphabetic or numeric literals. Alphabetic literals are denoted by enclosing them in single quotation marks. Operators fall into two main categories, arithmetic and logical. Arithmetic operators consist of + - / and * for addition, subtraction, division and multiplication respectively. The logical operators consist of EQ, NE, GT, LT, GE, LE, OR, and AND for equal, not equal, greater than, less than, greater

than or equal, less than or equal, or, and and. All logical operators are both preceded and followed by a period to separate them from operands. Standard FORTRAN priorities are used for operators to help clarify potentially ambiguous situations. Two additional operators are available, "(" and ")" to help when necessary. The researcher is restricted from using two different types of operands with the same operator. Numeric operands can be used with the following operators, +, -, *, /, EQ, NE, GT, LT, GE, LE. For example, the following statement could be used to calculate the average income of Ontario residents whose age is 70 years or more:

```
IF(PROV.EQ.'ONT'.AND.AGE.GE.70)CALCULATE AVERAGE
SALARY
```

Coded and alphabetic operands can use the operators EQ and NE. LOGICAL operators AND and OR are used to tie parts of the logical expression together. Coded and alphabetic literals enclosed in quotations should be the exact same length as the other operand they are connected to.

Some of the error messages that can be generated when this statement is scanned are as follows:

1) IF STATEMENT FOUND FOLLOWING ANOTHER IF

2) EXPECTING (AFTER IF - NOT FOUND

3) IMPROPER OR MISSING OPERATORS/OPERANDS IN IF

4) SYNTAX ERROR _ IMPROPER OPERATOR/OPERAND ORDER

5) UNEVEN BRACKETS

6) UNDECODEABLE LOGICAL/COMPARATIVE OPERATOR AFTER .

7) LOGICAL OPERATOR NOT FOLLOWED BY .

8) INVALID LOGICAL OPERATOR FOLLOWING .

9) MIXED MODE - TWO OPERANDS ARE NOT THE SAME TYPE

10) ALPHABETIC OR CODED OPERANDS WITH NON-ALPHA OPERATOR

11) NUMERIC OPERANDS WITH NON-NUMERIC OPERATOR

12) LOGICAL OPERANDS WITH NON-LOGICAL OPERATOR

13) TWO CODED VARIABLES AROUND EQ OR NE ARE NOT THE SAME LENGTH

3.2.3 Comment on Examples

This now completes the description of all of the instructions available in the tabulation section, along with approximately half of the error and warning messages available as diagnostic aids to the researcher. Appendix A contains a sample program that contains deliberate errors to show the error checking capability of the interpreter. Appendix B contains a working program that shows some of the capabilities of the language. The problem solved by the program is a very common one for almost all computer centres, a routine analysis of accounting data produced by programs run on a typical day. The data used pertains to the System/370 installation at the University of Toronto. The centre's User Services staff feels it needs to know the distribution of jobs at each of the more than ten remote

terminals that feed the main system. They need not only the number of jobs from each remote but specific information about the individual jobs as well. The system measurement group also keeps records on resource utilization. With the program in Appendix B, all the information that is desired is available.

One final item should be noted with regard to the results from the tabulation section. All results are output automatically in a fixed format. The nature of the language and its objectives does not lend itself to burdening the researcher with having to specify elaborate formatting schemes. Each line of a program is listed by the interpreter with a line number. All results generated by an instruction in the tabulation section refer back to the line number of the instruction that generated it. Although this is not the most elegant method, it is functional and easy to use.

LANGUAGE IMPLEMENTATION

4.1 MANDARIN - Manitoba Data Report Interpreter

4.1.1 Philosophy of Implementation

A great deal of thought went into the decisions on how to implement the language defined. The major decision to be made was whether to implement it as a compiler-based system or an interpretive system. There were definite advantages to both.

A compiler-based system was better for applications that would be performed repetetively. A load module could be produced once and this module could then be used to do the analysis. The compiler-based system was more efficient. There was no scanning needed once an object module was produced and the object module itself would be in machine or assembly code.

An interpretive system had other advantages. Users, especially novice ones, were not bothered with object modules and load modules. A system written in a high-level language that executed its own code was one step closer to being machine independent. If it was anticipated that the language would not be used for a production-like purpose then the overhead incurred could easily be balanced by the flexibility gained.

The solution that was eventually arrived at involved a combination of the advantages of both the

compiler-based and interpretive systems. All programs are scanned for correct syntax and an internal code is produced for the interpreter. It is this simplified code that is executed, eliminating many of the problems and much of the overhead of conventional systems. Both the translator and interpreter have been written in a high-level language and although not totally machine independent are as close as can be.

4.1.2 Gross Logic

The system is designed to be a batch processing system, that is many jobs could be processed and run with one loading of the interpreter.

Every job goes through four phases while being processed. The phases are individual processors that are necessary for translating the user's program into a form that is needed to manipulate his data.

A CONTROL PROCESSOR is used to act as a supervisor, passing control between the other processors. The FILE PROCESSOR handles all set-up requirements for I/O and the TABULATOR PROCESSOR looks after executable statements. Once all of the statements have been translated, control passes to the EXECUTION PROCESSOR for execution.

These are now discussed in greater detail.

4.1.3 MANDARIN ROUTINES

4.1.3.1 CONTROL PROCESSOR

A control section was developed to differentiate between different jobs of the same batch as well as sections of the same job. Control cards were implemented that indicated the different stages of processing that a user would go through. The control cards are:

1) JOB
2) FILE
3) TABULATE
4) DATA
5) EJECT

and 6) COMMENT

Two other control cards were added as aids to debugging the interpreter:

1) DEBUGON

and 2) DEBUGOFF

Every control card must be preceded by a control character in the first column of the card. This is a reserved character and may not be used for any other purpose if it is in column 1. All control cards take the form

control character control statement

In the current version of the interpreter @ is used as the control character. Appendix A and B contain examples of its use. The comment statement is slightly different. Instead of specifying

@COMMENT

for a comment card, the user only has to specify two control characters, @@.

Each control card has a very definite function. The JOB card is used to separate jobs in a batch as well as pass additional information on to the interpreter. When encountered, the CONTROL PROCESSOR passes control to routines that clean up any previous job in the same batch. Another routine is then performed that initializes data for the new job and picks up job options from the job card. These include the ability to suppress warning messages from the program listing and request the use of 2 byte integers for space saving in memory. The user requests these options by specifying keywords anywhere after the word JOB. The keywords for suppression of warning messages and the use of a smaller word size are NOWARN and SMALL respectively. The user's name and/or identification, if enclosed in single quotation marks, is also picked up from this card and is used when a job separator page is produced. Following the production of the separator page and variable initialization, control returns to a main input routine to get the next card.

The FILE card is usually found following the job card. When encountered the CONTROL PROCESSOR initializes variables needed in the definition of I/O files and returns to the main input routine.

The TABULATE card invokes routines that terminate the previous processor. Variables needed for the TABULATE PROCESSOR are initialized and control returns again to the main input routine.

The DATA card signifies that all commands or instructions have been entered. Internal code is generated to provide a "flow control" through the code that was previously generated. Variables needed in execution are initialized and if no errors have been recorded to this point, control passes to the EXECUTION PROCESSOR.

The fifth control card is the EJECT card. This is only used to aid in the ouput listing of a program. When encountered, the printer is forced to skip to the top of a new page where the listing is continued.

The COMMENT card is a convenient method of supplying comments or documentation in a program. All comments are listed in the order they are found and have no other effect upon a program.

The DEBUGON and DEBUGOFF cards are used to selectively turn on and off special procedures used to

debug the interpreter. They are of very limited use to the programmer but can be of significant help to the systems personnel responsible for the interpreter. The backbone of the debugging procedures is a formatted dump routine that prints out main memory for the interpreter along with some of the important pointers and variables.

In addition to providing the control between each of these control sections, the CONTROL PROCESSOR does a certain amount of error checking to insure that program sections are in the proper sequence and one program does not destroy another one in the same batch.

4.1.3.2 FILE PROCESSOR

A file section handles the formation of data structures needed by the interpreter as well as the allocation of storage space for variables defined by the user.

Control is passed from the main input routine to the analysis portion of the file section. If the card just input indicates the beginning of a new file (i.e. a FILENAME card) the last file is closed off and the new file is stored in the interpreter's memory. All files are linked together to provide the capability for efficient file searches. Checks are provided to insure that the user does not try to define the same file more than once.

If the card passed to the file section is not a FILENAME card it is assumed to be a variable definition card. Routines are called that break this input card into syntactic units. If the syntax of the statement is correct the data that has been broken down is used to build the information about the variable that will be needed at execution time. The FILE PROCESSOR links all of the variables of a program together for efficient variable searches. As with the file definitions, checks are made to insure unique variable names in the definition phase.

4.1.3.3 TABULATE PROCESSOR

The third section is the tabulation section. Its function is to do all syntax checking of executable statements, issue error and warning messages when necessary, and generate the internal text for the interpreter.

In many respects, the TABULATE PROCESSOR is similar to the FILE PROCESSOR. Statements passed to the processor have their first syntactic unit used in a table look-up sequence. If the unit matches any element in the table, control is passed to a specialized routine that continues syntax checking and internal code generation. The specialized routines use the same code as the FILE PROCESSOR for doing the syntax analysis and error message generation. If a statement is

syntactically correct, its equivalent internal code is loaded into memory. All variable references are resolved and run-time error checking is included if needed. Control returns to the main input routine so that the next statement may be read in. The TABULATE PROCESSOR also keeps track of the memory location of the first statement encountered and this is passed on to the EXECUTION PROCESSOR.

4.1.3.4 EXECUTION PROCESSOR

The final processor is the interpreter. It is responsible for executing the code produced by the TABULATE PROCESSOR and for issuing execution-time error messages. The interpreter also keeps track of certain job statistics such as memory utilization, the number of records read from an input file or written to an output file, and the number of error and warning messages generated for the job.

The EXECUTION PROCESSOR uses three main parts of the interpreter, an Instruction Address Register, a branch table and instruction modules. The Instruction Address Register (IAR) is initially set to point to the first instruction in memory. This is passed to the EXECUTION PROCESSOR from the TABULATE PROCESSOR. The EXECUTION PROCESSOR picks up the byte in memory pointed to by the IAR and uses this as an index into a branch table. The byte of information actually represents an

operation code of a specific instruction. The branch table contains pointers to the different routines needed to execute the different operation codes. When control passes to these routines, they use the information following the operation code as operands. After processing the operation code, the IAR is incremented by a fixed number of bytes so that it points to the next operation code. Control passes back to the point where the next operation code is picked up and used in the branch table.

Each instruction (operation code) has two phases. READ and WRITE instructions are executed completely each time they are encountered in a looping program. This is what one would normally expect. A statement requesting the AVERAGE value of a field cannot be fully executed until all of the data has been collected. The two phases for each instruction could be called the "loop phase" and the "final output phase". For READ and WRITE instructions, a READ or WRITE is performed every time it is encountered in the loop phase. CALCULATE instructions only collected data during the "loop phase". The loop phase ends and the "final output phase" begins when all the data has been collected, that is when an end-of-file interrupt is raised while doing a READ operation. At this point one final circuit is made through the program, not to add new data to totals or averages but

to complete the final calculations and output the results. This is the "final output phase".

4.2 Internal Organization

4.2.1 Symbol Table Entries

The symbol table is located at the low end of the interpreter's memory. It contains the internal format of all of the files defined in the tabulation section of a program. All variables defined are entered into the table and are connected to all previously defined variables in a linked list structure. This allows for efficient searches of the symbol table while allowing for variable length entries in the table. In addition to a link each entry in the table contains the variable name, and displacement from the beginning of the record for which the data item is defined, the length of the data item, its type (numeric, coded or alphanumeric), a pointer to a memory location where the description of the variable is located if the description has been included, and a pointer to the coded description if the variable is of the coded type. Additional space has been included for possible future expansion but is not presently used.

4.2.2 Instruction Entries

The following sections describe briefly the internal code that is generated by the different processors mentioned above. Each statement from a

program is broken down into an operation code and operands.

4.2.2.1 READ

The READ instruction takes the following format:

OP CODE,FILE-NUMBER-POINTER

The File-Number-Pointer points to a memory location where the file structure for the file in question is located.

4.2.2.2 WRITE

The WRITE instruction takes the identical format to that of the READ.

OP CODE,FILE-NUMBER-POINTER

The pointer indicates the area defining the file structure and data to be output.

4.2.2.3 MOVE

The MOVE instruction has the form

OP CODE,POINTER 1,POINTER 2,Length,
Padding (if necessary)

Pointer 2 points to the information to be moved while Pointer 1 points to the area into which the data is to move. Length represents the number of characters to be transferred and padding represents the number of blanks to be used to fill out a field that is larger than need be.

4.2.2.4 CALCULATE

The CALCULATE statement is really three separate statements, one for SUM, AVERAGE and STANDARD DEVIATION. Each consists of the following:

OP CODE,CODE-COUNTER,DATA-POINTERS,
ACCUMULATORS

The code-counter is used when a CALCULATE statement is used with a "BY CODED variable" phrase as in the following:

CALCULATE AVERAGE AGE BY CITY

where CITY is coded.

The code-counter contains the dimensions of the variable CITY (i.e. the number of codes that were defined for that variable). The data-pointers point to the memory locations of the CODED variable and the data to be used in the calculations. The value of the code-counter specifies the number of sets of accumulators needed for the computation. If, in the above example, seven cities were defined, we would need seven sets of accumulators, one for each city. The following table indicates the number of accumulators in each set for each CALCULATE statement:

Statement	# of accumulators
SUM/TOTAL	1
AVERAGE	2
STANDARD DEVIATION	3

If a CODED variable is not used, one set of accumulators is sufficient for all calculations.

4.2.2.5 ISN

The ISN instruction is generated at the beginning of every program statement. It takes the form

OP CODE,LINE-NUMBER

The line-number refers to the statement number of the statement about to be processed. When encountered during execution, this statement updates an instruction counter that can be used in execution-time error messages and diagnostics.

4.2.2.6 CLASSIFY/ORDER

CLASSIFY and ORDER are treated in the same section because of the similarity of their organization and the way in which they are tied together. The CLASSIFY instruction takes the following form:

OP CODE,VARIABLE-COUNT,VECTOR-SIZE,VARIABLE-POINTERS,DIMENSIONS,MULTIPLIERS,COUNTER-VECTOR

VARIABLE-COUNT (n) is the number of variables in the statement. VECTOR-SIZE (m) is the product of the dimensions of the n variables. VARIABLE-POINTERS point to the memory locations of the n variables. DIMENSIONS represents the n dimensions of the n variables, and MULTIPLIERS (n of them) are used for indexing into the vector of counters and are defined as follows:

```
MULTIPLIER(N) = 1
MULTIPLIER(N-1) = DIMENSION(N)* MULTIPLIER(N)
```

COUNTERS (m of them) are used to count the occurrence at each possible combination of variables.

The ORDER statement has the following form:

OP CODE,VARIABLE-POINTERS,DIMENSIONS, MULTIPLIERS

All of these have the same meaning as the CLASSIFY with one exception. If variable i in the order statement has the position j in the previous CLASSIFY, multiplier i in the ORDER instruction takes the value of multiplier j from the previous CLASSIFY instruction. When these multipliers are used to index into the vector from the previous CLASSIFY they will access the same data in a new order.

4.2.2.7 IF

The IF instruction follows a very simple logic and takes the form:

OP CODE,EXPRESSION-RESULT-POINTER,NEW-INSTRUCTION-ADDRESS

The Expression-Result-Pointer points to an area of memory that contains a true or false value (1 or 0) representing the validity of the logical expression originally specified in the IF statement of the program. If the statement was true, the Instruction Address Register (IAR) is set to point to the next instruction in memory. If the statement was false, the IAR is assigned the New-Instruction-Address from the IF instruction. This new address is just the location

after the instruction that followed at the end of the IF statement in the program.

4.2.2.8 PLOT

There are two forms of the PLOT statement, one for CODED data and one for NUMERIC.

The CODED PLOT takes the following form:

OP CODE,DIMENSION,VARIABLE-ADDRESS,COUNTERS

DIMENSION(n) is the number of entries for the CODED variable. VARIABLE-ADDRESS points to the memory location defining the variable being plotted. COUNTERS represents the n accumulators needed for the n bars of the histogram plot.

The NUMERIC PLOT takes the following form:

OP CODE,GROUPS,VARIABLE-ADDRESS,RANGE-1,
RANGE-2, COUNTERS

GROUPS represents the number of segments that are to be plotted in the numeric span between RANGE-1 and RANGE-2. VARIABLE-ADDRESS points to the variable definition of the data being plotted and COUNTERS represents the n accumulators needed for the n bars of the histogram plot.

Each of the above instructions was designed to use as little memory space as possible but still provide the necessary information for efficient execution.

Conclusion

In trying to evaluate this thesis the author has asked four basic questions:

1) Does the implementation of the language follow the design specifications and work as it should?

2) Does the project meet the objectives originally proposed?

3) With the aid of hindsight, how would the project have been changed to possibly attain greater results?

4) Was the project a success?

The implementation of the language follows the specifications exactly and does indeed work. Extensive testing was performed and the interpreter has no known "bugs". By far, the largest number of problems that arose during final testing of the interpreter were due to keypunch errors in the programs being tested or the data being used. These were all caught by the interpreter and the diagnostics that were generated made the error correction trivial.

There were four basic objectives originally considered and most of them have been met. Simplicity has been maintained throughout. The language is restricted to a few basic, powerful commands that are easy to use but still do the work. The language seems to have a wide range of applications even in a production environment for daily statistical applications as can be

seen in the example in the appendix. Flexibility has been maintained from the language definitions through to the interpreter implementations. Options have been provided to make the use of the language as simple as possible. The final objective, generality, has not been completely attained. Even though the interpreter was written in a high-level language a few sections of code were written in a machine-dependent manner. These sections are very few in number and should not produce many problems if a version is required for another machine (i.e. other than IBM/360 or 370).

Two design decisions would be different if the project was being re-written. One would be to write specific parts of the translator in a low-level language for greater efficiency. The second would be to write a compiler version instead of an interpreter. Originally when this language was still in the design stage it was felt that its main uses would be as a pre-statistical package data analyzer. A researcher with his data would run a program a few times to determine whether or not to go on and do an in-depth analysis. Since the interpreter has been finished more and more applications have developed that can use the language as it is and do not need anything else. For these applications that are run on a regular basis, an interpreter does not offer the efficiency that is desired and can only be accomodated by a load module.

The project can definitely be termed a success for the following two reasons:

1) The project met or exceeded almost every objective.

2) The author has gained a great deal of knowledge in the areas of language design, language implementation and large-project implementation.

APPENDIX A

This section contains two sample jobs with deliberate programming errors. It is meant to show the diagnostic features of the interpreter.

```
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER
MANDARIN    VERSION 0.1    COMPILED FEB. 28,1972.    JOB LAWRIE-BUG TEST1    MANITOBA DATA REPORT INTERPRETER

                                   J O B   O P T I C N S

                                         WARN

                                         CTHER

                                         SMALL
```

```
      @FILE                                                                    00000240
    1   FILENAME(FILE05)                                                       00000250
    2     NAME=COL(1-8),A,'FAMILY NAME'                                        00000260
    3     VISITS=COL(18-19),N                                                  00000270
    4     PAID=COL(26-30),N,'PAYMENTS MADE BY CLIENTS'                         00000280
    5     OWED=COL(31-35),N,'PAYMENTS OWED BY CLIENTS'                         00000290
    6     CITY=COL(39-41),C,(TOR,MON,QUE,WIN,VAN)                              00000300
      @TABULATE                                                                00000310
    7   READ(FILE05)                                                           00000320
    8   CALCULATE                                                              00000330
***ERROR*** EXPECTING ARITHMETIC FUNCTION AFTER "CALCULATE" BUT IT WAS NOT FOUND.
    9   CALCULATE TOTAL PAID BY CITY                                           00000340
   10   CALCULATE TOTAL PAID BY VISITS                                         00000350
***ERROR*** CODED VARIABLE EXPECTED AFTER "BY" BUT  NUMERIC OR ALPHABETIC FOUND
   11   CALCULATE AVERAGE AGE                                                  00000360
***ERROR*** A VARIABLE IN THE ABOVE STATEMENT DID NOT APPEAR  IN THE FILE SECTION.
   12   IF)OWED.GT.50.AND.CITY.EQ.'TOR')CALCULATE AVERAGE PAID                 00000390
***ERROR*** EXPECTING ( AFTER IF -NOT FOUND
   13   PLOT BY VISITS                                                         00000400
***ERROR*** EXPECTING "GROUP" OR "RANGE" SPECIFICATION BUT NOT FOUND
   14   PLOT BY CITY                                                           00000410
   15   PLOT VISITS GROUPS=RANGE=10                                            00000420
***ERROR*** EXPECTING NUMERIC VALUE AFTER GROUPS.
      @DATA                                                                    00000430
        EXECUTION STEP NOT TAKEN DUE TO PREVIOUS ERRORS.
***ERROR*** EOJ STATEMENT MISSING.IT HAS BEEN ADDED.

CORE USAGE          OBJECT CODE AND VARIABLE STORAGE=   173 WORDS, TOTAL AVAILABLE=  1500 WORDS

DIAGNOSTICS         NUMBER OF ERRORS=     7, NUMBER OF WARNINGS=     0

NO RECORDS WERE INPUT OR OUTPUT DURING THIS EXECUTION
```

```
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-BUG TEST2     MANITOBA DATA REPORT INTERPRETER

                                     J O B   O P T I O N S

                                         WARN

                                         OTHER

                                         SMALL
```

```
        @FILE                                                                   00000070
   1          FILENAME)FILE05)                                                  00000080
***ERROR*** IMPROPER CHARACTER FOLLOWING FILENAME.
   2          FILENAME(FILE95)                                                  00000090
***ERROR*** FILE NUMBER OUTSIDE RANGE 1-20.
   3          FILENAME(FILE15)                                                  00000100
   4             AGE1=COL(27-28),NUMERIC,'AGE OF PATIENT'                       00000110
   5             HT1=COL(34-36),NUMERIC,'HEIGHT IN INCHES'                      00000120
   6             NAME=CCLUNMS(37-46),ALFA,'PATIENT SURNAME'                     00000130
**WARNING** "CCLUMN" EXPECTED BUT MISSPELLED CORRECTION HAS BEEN EFFECTED.
**WARNING** ALPHABETIC TYPE EXPECTED BUT MISSPELLED.
             CORRECTION HAS BEEN EFFECTED.
   7     9  FILENAME(FILE06)                                                  00000140
***ERROR*** FILE OR RECORD SPECIFICATION MUST START WITH AN ALPHABETIC CHARACTER.
   8             NAME=CCLUMNS(37-46),ALPHABETIC,'PATIENT SURNAME'               00000150
***ERROR*** VARIABLE NAME HAS BEEN PREVIOUSLY DEFINED. THIS REFERENCE IS AMBIGUOUS.
**WARNING** VARIABLES HAVE NOT BEEN SPECIFIED IN COLUMN SEQUENCE.
   9             AGE2=COL(50-51),N                                              00000160
        @TABULATE                                                               00000161
        @DATA                                                                   00000162
           EXECUTION STEP NOT TAKEN DUE TO PREVIOUS ERRORS.
        @ECJ                                                                    00000165

CORE USAGE          OBJECT CODE AND VARIABLE STORAGE=    116 WORDS, TOTAL AVAILABLE=  1500 WORDS

DIAGNOSTICS         NUMBER OF ERRORS=      4, NUMBER OF WARNINGS=      3

NO RECORDS WERE INPUT OR OUTPUT DURING THIS EXECUTION
```

APPENDIX B

This section contains a sample working job. It not only shows that the interpreter works but also shows some of its functional capability. The program shown solves a problem many computer centres have. Given accounting records from jobs run, do a statistical analysis on certain aspects of the data. The University of Toronto has a very large centre with many small terminals scattered throughout the different campuses that act as feeders to the central site. Management likes to keep a profile of the types of jobs that run on the system, that is, where they come from, what priority do they use, cards read in, lines printed, etc. The file section of the example defines the structure of one of the accounting records. The tabulate section contains the instructions for the analysis. The first three instructions generate plotted output. The next eight instructions calculate required statistics and yield a numerical output. The CLASSIFY and ORDER statements generate the bulk of the output and count the occurrences of the different combinations of job options available at the centre. The final statement counts the number of forms mounts of jobs run priority 8 and submitted through the terminal system.

```
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER
MANDARIN     VERSION 0.1     COMPILED FEB. 28,1972.     JOB LAWRIE-ACCTG SUMMARY     MANITOBA DATA REPORT INTERPRETER

                                              J O B   O P T I O N S

                                                  WARN

                                                  OTHER

                                                  SMALL
```

```
    1     FILENAME(FILE05)                                                                00000030
    2       JOBNAM=COL(37-44),ALPHA,'OS JCB NAME'                                         00000040
**WARNING** ALPHABETIC TYPE EXPECTED BUT MISSPELLED.
            CORRECTION HAS BEEN EFFECTED.
    3       RTCODE=COL(37 -38),CODED,'REMOTE TERMINAL CODE',(CC/CENTRE,NP/PHYSICS-        00000050
                ,AS/ARTS,SC/SCARBOROUGH,ER/ERINDALE,DE/DENTISTRY,AE/AEROSPACE  -          00000060
                ,SB/BUSINESS,UW/WESTERN1,WS/WESTERN2,CR/CRJE,TS/TSO,EU/BATCH)             00000070
**WARNING** VARIABLES HAVE NOT BEEN SPECIFIED IN COLUMN SEQUENCE.
    4       CPUTIM=COLS(45-48),NUMERIC,'CPU TIME-10THS OF MIN'                            00000080
    5       CORE=COLS(50-55),NUMERIC,'CORE USAGE-KB MIN'                                  00000090
    6       CARDIN=COLS(57-61),NUMERIC,'CARDS READ'                                       00000100
    7       LINES=COLS(62-67),NUMERIC,'LINES PRINTED'                                     00000110
    8       PUNCH=COLS(68-72),NUMERIC,'CARDS PUNCHED'                                     00000120
    9       CLASS=COL(73),CODED,'HASP/OS JCB CLASS',(A,B,C,M,W)                           00000130
   10       PRIOR=COLS(74-75),CODED,'JCB PRIORITY',(04,06,08,00)                          00000140
   11       DISKS=COL(76),NUMERIC,'# DISK MOUNTS'                                         00000150
   12       TAPES=COL(77),NUMERIC,'# TAPE MOUNTS'                                         00000160
   13       FORMS=COL(78),NUMERIC,'# FORMS MOUNTS'                                        00000170
   14       PUNMT=COL(79),NUMERIC,'# PUNCH MOUNTS'                                        00000180
   15       VERSION=COL(80),ALPHA                                                         00000190
**WARNING** VARIABLE NAME IS TOO LONG. IT HAS BEEN TRUNCATED TO 6 CHARACTERS
**WARNING** ALPHABETIC TYPE EXPECTED BUT MISSPELLED.
            CORRECTION HAS BEEN EFFECTED.
         @TABULATE                                                                        00000200
   16            READ(FILE05)                                                           00000210
   17            PLOT BY RTCODE                                                         00000220
   18            PLOT BY PRIOR                                                          00000230
   19            PLOT BY CLASS                                                          00000240
   20                 CALCULATE TOTAL LINES                                             00000250
   21                 CALCULATE TOTAL LINES BY RTCODE                                   00000260
   22                 CALCULATE AVERAGE LINES                                           00000270
   23                 CALCULATE TOTAL CARDIN                                            00000280
   24                 CALCULATE TOTAL CARDIN BY RTCODE                                  00000290
   25                 CALCULATE AVERAGE CARDIN                                          00000300
   26                 CALCULATE TOTAL CPUTIM                                            00000310
   27                 CALCULATE AVERAGE CPUTIM                                          00000320
   28            CLASSIFY(RTCODE,CLASS,PRIOR)                                           00000330
   29              ORDER(RTCODE,PRIOR,CLASS)                                            00000340
   30              ORDER(PRIOR,RTCODE,CLASS)                                            00000350
   31              ORDER(PRIOR,CLASS,RTCODE)                                            00000360
   32              ORDER(CLASS,RTCODE,PRIOR)                                            00000370
   33              ORDER(CLASS,PRIOR,RTCODE)                                            00000380
   34            IF(CLASS.EQ.'C'.AND.LINES.GT.15000)PLOT BY RTCODE                      00000390
         @DATA                                                                          00000400
```

```
        MINIMUM                                                                                              MAXIMUM
              0                                                                                                 1467

CODE/DESCR. OR RANGE                    NUMBER OF OCCURANCES
REMOTE TERMINAL CODE------------------------------------------------------------------------------------------------
                      |
                      |
                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
CC/CENTRE             |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=       515
                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                      |
                      |
                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
NP/PHYSICS            |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=       503
                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                      |
                      |
                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
AS/ARTS               |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=      1467
                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                      |
                      |
                      |XXXXX
SC/SCARBOROUGH        |XXXXX #=       92
                      |XXXXX
                      |
                      |
                      |X
ER/ERINDALE           |X #=       30
                      |X
                      |
                      |
CE/DENTISTRY          |  #=       0
                      |
                      |
                      |
                      |XXX
AE/AEROSPACE          |XXX #=       63
                      |XXX
                      |
                      |
                      |XX
SB/BUSINESS           |XX #=       34
                      |XX
                      |
                      |
                      |
UW/WESTERN1           |  #=       2
                      |
                      |
                      |
                      |
WS/WESTERN2           |  #=       0
                      |
                      |
                      |
```

```
CR/CRJE         |XXXXXXXXXXXXX #=      219
                |XXXXXXXXXXXXX
                |
                |
                |XXX
TS/TSO          |XXX #=      50
                |XXX
                |
                |
                |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
EU/BATCH        |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=      982
                |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                |
                |
```

```
           MINIMUM                                                                                              MAXIMUM
                 0                                                                                                 2839
CCDE/DESCR. OR RANGE                    NUMBER OF CCCURANCES
JCB PRIORITY            ------------------------------------------------------------------------------------------------
                        |
                        |
                        |XXXXXXXXXXXXXXXXXXXX
04                      |XXXXXXXXXXXXXXXXXXXX #=       619
                        |XXXXXXXXXXXXXXXXXXXX
                        |
                        |
                        |XXXXXXXXXXXXXXXXX
06                      |XXXXXXXXXXXXXXXXX #=       546
                        |XXXXXXXXXXXXXXXXX
                        |
                        |
                        |XXXXXXX
08                      |XXXXXXX #=       227
                        |XXXXXXX
                        |
                        |
                        |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
00                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=       2839
                        |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                        |
                        |
```

```
           MINIMUM                                                                                                      MAXIMUM
                0                                                                                                          2791
CODE/DESCR. OR RANGE                      NUMBER OF OCCURANCES
HASP/OS.JOB CLASS      -------------------------------------------------------------------------------------------
                       |
                       |
                       |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
A                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=       935
                       |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                       |
                       |
                       |XXXXXX
B                      |XXXXXX #=       196
                       |XXXXXX
                       |
                       |
                       |XXX
C                      |XXX #=       106
                       |XXX
                       |
                       |
                       |XXXXX
M                      |XXXXX #=       152
                       |XXXXX
                       |
                       |
                       |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
W                      |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX #=      2791
                       |XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                       |
                       |
NAME         DESCRIPTION                             THIS DATA IS GENERATED BY LINE  20
LINES        LINES PRINTED
THE SUM OF LINES. IS   2028375

NAME         DESCRIPTION                             THIS DATA IS GENERATED BY LINE  21
LINES        LINES PRINTED
THE STATISTICS GIVEN ARE CALCULATED FOR EACH GROUP OF THE FOLLOWING CODED VARIABLE
NAME         DESCRIPTION
RTCODE       REMOTE TERMINAL CODE
(  1)     CC    CENTRE
(  2)     NP    PHYSICS
(  3)     AS    ARTS
(  4)     SC    SCARBOROUGH
(  5)     ER    ERINDALE
(  6)     DE    DENTISTRY
(  7)     AE    AEROSPACE
(  8)     SB    BUSINESS
(  9)     UW    WESTERN1
( 10)     WS    WESTERN2
( 11)     CR    CRJE
( 12)     TS    TSO
( 13)     EU    BATCH
( 14)     NONE OF THE ABOVE
ANY OF THE ABOVE CATEGORIES NOT LISTED BELOW HAD NO ENTRIES
```

```
THE SUM OF LINES  WITHIN CATEGORY (  2) IS     193635
THE SUM OF LINES  WITHIN CATEGORY (  3) IS     400723
THE SUM OF LINES  WITHIN CATEGORY (  4) IS      20129
THE SUM OF LINES  WITHIN CATEGORY (  5) IS      10490
THE SUM OF LINES  WITHIN CATEGORY (  6) IS          0
THE SUM OF LINES  WITHIN CATEGORY (  7) IS       8627
THE SUM OF LINES  WITHIN CATEGORY (  8) IS      27886
THE SUM OF LINES  WITHIN CATEGORY (  9) IS      11484
THE SUM OF LINES  WITHIN CATEGORY ( 10) IS          0
THE SUM OF LINES  WITHIN CATEGORY ( 11) IS     160173
THE SUM OF LINES  WITHIN CATEGORY ( 12) IS      17057
THE SUM OF LINES  WITHIN CATEGORY ( 13) IS     271462
THE SUM OF LINES  WITHIN CATEGORY ( 14) IS     350459

NAME          DESCRIPTION                    THIS DATA IS GENERATED BY LINE  22
LINES         LINES PRINTED
THE AVERAGE OF LINES  IS      479.4

NAME          DESCRIPTION                    THIS DATA IS GENERATED BY LINE  23
CARDIN        CARDS READ
THE SUM OF CARDIN IS     997334

NAME          DESCRIPTION                    THIS DATA IS GENERATED BY LINE  24
CARDIN        CARDS READ
THE STATISTICS GIVEN ARE CALCULATED FOR EACH GROUP OF THE FOLLOWING CODED VARIABLE
NAME          DESCRIPTION
RTCODE        REMOTE TERMINAL CODE
(  1)     CC     CENTRE
(  2)     NP     PHYSICS
(  3)     AS     ARTS
(  4)     SC     SCARBOROUGH
(  5)     ER     ERINDALE
(  6)     DE     DENTISTRY
(  7)     AE     AEROSPACE
(  8)     SB     BUSINESS
(  9)     UW     WESTERN1
( 10)     WS     WESTERN2
( 11)     CR     CRJE
( 12)     TS     TSO
( 13)     EU     BATCH
( 14)     NONE OF THE ABOVE
ANY OF THE ABOVE CATEGORIES NOT LISTED BELOW HAD NO ENTRIES
THE SUM OF CARDIN WITHIN CATEGORY (  1) IS     209208
THE SUM OF CARDIN WITHIN CATEGORY (  2) IS     155281
THE SUM OF CARDIN WITHIN CATEGORY (  3) IS     284242
THE SUM OF CARDIN WITHIN CATEGORY (  4) IS       7548
THE SUM OF CARDIN WITHIN CATEGORY (  5) IS      15127
THE SUM OF CARDIN WITHIN CATEGORY (  6) IS          0
THE SUM OF CARDIN WITHIN CATEGORY (  7) IS      15819
THE SUM OF CARDIN WITHIN CATEGORY (  8) IS      11983
THE SUM OF CARDIN WITHIN CATEGORY (  9) IS       3768
THE SUM OF CARDIN WITHIN CATEGORY ( 10) IS          0
THE SUM OF CARDIN WITHIN CATEGORY ( 11) IS      52362
THE SUM OF CARDIN WITHIN CATEGORY ( 12) IS       4373
THE SUM OF CARDIN WITHIN CATEGORY ( 13) IS     202137
THE SUM OF CARDIN WITHIN CATEGORY ( 14) IS      35486

NAME          DESCRIPTION                    THIS DATA IS GENERATED BY LINE  25
CARDIN        CARDS READ
```

NAME DESCRIPTION THIS DATA IS GENERATED BY LINE 26
CPUTIM CPU TIME-10THS OF MIN
THE SUM OF CPUTIM IS 2525

NAME DESCRIPTION THIS DATA IS GENERATED BY LINE 27
CPUTIM CPU TIME-10THS OF MIN
THE AVERAGE OF CPUTIM IS 0.6

RTCODE	CLASS	PRIOR	OCCURENCES	SUBTOTALS
CENTRE	A	04	215	
CENTRE	A	06	127	
CENTRE	A	08	41	
CENTRE	A	00	0	
CENTRE	A	CTHERS	0	
				383
CENTRE	B	04	34	
CENTRE	B	06	27	
CENTRE	B	08	22	
CENTRE	B	00	0	
CENTRE	B	CTHERS	0	
				83
CENTRE	C	04	10	
CENTRE	C	06	24	
CENTRE	C	08	3	
CENTRE	C	00	0	
CENTRE	C	CTHERS	0	
				37
CENTRE	M	04	0	
CENTRE	M	06	0	
CENTRE	M	08	0	
CENTRE	M	00	0	
CENTRE	M	CTHERS	0	
				0
CENTRE	W	04	0	
CENTRE	W	06	0	
CENTRE	W	08	0	
CENTRE	W	00	0	
CENTRE	W	CTHERS	0	
				0
CENTRE	CTHERS	04	0	
CENTRE	CTHERS	06	0	
CENTRE	OTHERS	08	0	
CENTRE	CTHERS	00	12	
CENTRE	CTHERS	CTHERS	0	
				12
PHYSICS	A	04	126	
PHYSICS	A	06	52	
PHYSICS	A	08	1	
PHYSICS	A	00	2	
PHYSICS	A	CTHERS	0	
				181
PHYSICS	B	04	9	
PHYSICS	B	06	1	
PHYSICS	B	08	2	
PHYSICS	B	00	0	
PHYSICS	B	CTHERS	0	
				12
PHYSICS	C	04	2	
PHYSICS	C	06	1	
PHYSICS	C	08	0	
PHYSICS	C	00	0	
PHYSICS	C	CTHERS	0	
				3
PHYSICS	M	04	0	
PHYSICS	M	06	0	
PHYSICS	M	08	0	

PHYSICS	M	CTHERS	0
			0
PHYSICS	W	04	0
PHYSICS	W	06	0
PHYSICS	W	08	0
PHYSICS	W	00	301
PHYSICS	W	CTHERS	0
			301
PHYSICS	OTHERS	04	1
PHYSICS	OTHERS	06	0
PHYSICS	OTHERS	08	0
PHYSICS	OTHERS	00	5
PHYSICS	OTHERS	OTHERS	0
			6
ARTS	A	04	75
ARTS	A	06	68
ARTS	A	08	8
ARTS	A	00	0
ARTS	A	CTHERS	0
			151
ARTS	B	04	9
ARTS	B	06	1
ARTS	B	08	2
ARTS	B	00	0
ARTS	B	CTHERS	0
			12
ARTS	C	04	2
ARTS	C	06	0
ARTS	C	08	0
ARTS	C	00	0
ARTS	C	CTHERS	0
			2
ARTS	M	04	0
ARTS	M	06	0
ARTS	M	08	0
ARTS	M	00	0
ARTS	M	CTHERS	0
			0
ARTS	W	04	0
ARTS	W	06	0
ARTS	W	08	0
ARTS	W	00	1287
ARTS	W	CTHERS	0
			1287
ARTS	OTHERS	04	1
ARTS	OTHERS	06	0
ARTS	OTHERS	08	0
ARTS	OTHERS	00	14
ARTS	OTHERS	OTHERS	0
			15
SCARBORCU	A	04	0
SCARBORCU	A	06	5
SCARBORCU	A	08	0
SCARBORCU	A	00	0
SCARBORCU	A	CTHERS	0
			5
SCARBORCU	B	04	4
SCARBORCU	B	06	7
SCARBORCU	B	08	0

SCARBCRCU	B	CTHERS	0
			11
SCARBCRCU	C	04	1
SCARBOROU	C	06	0
SCARBOROU	C	08	0
SCARBCRCU	C	00	0
SCARBOROU	C	CTHERS	0
			1
SCARBCRCU	M	04	0
SCARBOROU	M	06	24
SCARBOROU	M	08	0
SCARBCRCU	M	00	0
SCARBOROU	M	CTHERS	0
			24
SCARBCRCU	W	04	0
SCARBOROU	W	06	0
SCARBCRCU	W	08	0
SCARBCRCU	W	00	51
SCARBOROU	W	CTHERS	0
			51
SCARBCRCU	CTHERS	04	0
SCARBOROU	OTHERS	06	0
SCARBCRCU	OTHERS	08	0
SCARBCRCU	CTHERS	00	0
SCARBOROU	OTHERS	CTHERS	0
			0
ERINDALE	A	04	18
ERINDALE	A	06	2
ERINDALE	A	08	0
ERINDALE	A	00	0
ERINDALE	A	CTHERS	0
			20
ERINDALE	B	04	0
ERINDALE	B	06	0
ERINDALE	B	08	0
ERINDALE	B	00	0
ERINDALE	B	OTHERS	0
			0
ERINDALE	C	04	1
ERINDALE	C	06	0
ERINDALE	C	08	0
ERINDALE	C	00	0
ERINDALE	C	OTHERS	0
			1
ERINDALE	M	04	0
ERINDALE	M	06	0
ERINDALE	M	08	0
ERINDALE	M	00	0
ERINDALE	M	OTHERS	0
			0
ERINDALE	W	04	0
ERINDALE	W	06	0
ERINDALE	W	08	0
ERINDALE	W	00	9
ERINDALE	W	OTHERS	0
			9
ERINDALE	CTHERS	04	0
ERINDALE	CTHERS	06	0
ERINDALE	CTHERS	08	0

ERINDALE	OTHERS	OTHERS	0
			0
DENTISTRY	A	04	0
DENTISTRY	A	06	0
DENTISTRY	A	08	0
DENTISTRY	A	00	0
DENTISTRY	A	OTHERS	0
			0
DENTISTRY	B	04	0
DENTISTRY	B	06	0
DENTISTRY	B	08	0
DENTISTRY	B	00	0
DENTISTRY	B	OTHERS	0
			0
DENTISTRY	C	04	0
DENTISTRY	C	06	0
DENTISTRY	C	08	0
DENTISTRY	C	00	0
DENTISTRY	C	OTHERS	0
			0
DENTISTRY	M	04	0
DENTISTRY	M	06	0
DENTISTRY	M	08	0
DENTISTRY	M	00	0
DENTISTRY	M	OTHERS	0
			0
DENTISTRY	W	04	0
DENTISTRY	W	06	0
DENTISTRY	W	08	0
DENTISTRY	W	00	0
DENTISTRY	W	OTHERS	0
			0
DENTISTRY	OTHERS	04	0
DENTISTRY	OTHERS	06	0
DENTISTRY	OTHERS	08	0
DENTISTRY	OTHERS	00	0
DENTISTRY	OTHERS	OTHERS	0
			0
AEROSPACE	A	04	9
AEROSPACE	A	06	3
AEROSPACE	A	08	0
AEROSPACE	A	00	0
AEROSPACE	A	OTHERS	0
			12
AEROSPACE	B	04	0
AEROSPACE	B	06	0
AEROSPACE	B	08	0
AEROSPACE	B	00	0
AEROSPACE	B	OTHERS	0
			0
AEROSPACE	C	04	0
AEROSPACE	C	06	0
AEROSPACE	C	08	0
AEROSPACE	C	00	0
AEROSPACE	C	OTHERS	0
			0
AEROSPACE	M	04	0
AEROSPACE	M	06	0
AEROSPACE	M	08	0

AERCSPACE	M	OTHERS	0
			0
AERCSPACE	W	04	0
AERCSPACE	W	06	0
AEROSPACE	W	08	0
AERCSPACE	W	00	51
AERCSPACE	W	OTHERS	0
			51
AERCSPACE	OTHERS	04	0
AERCSPACE	OTHERS	06	0
AEROSPACE	OTHERS	08	0
AERCSPACE	OTHERS	00	0
AERCSPACE	OTHERS	OTHERS	0
			0
BUSINESS	A	04	14
BUSINESS	A	06	10
BUSINESS	A	08	0
BUSINESS	A	00	0
BUSINESS	A	OTHERS	0
			24
BUSINESS	B	04	0
BUSINESS	B	06	0
BUSINESS	B	08	1
BUSINESS	B	00	0
BUSINESS	B	OTHERS	0
			1
BUSINESS	C	04	5
BUSINESS	C	06	1
BUSINESS	C	08	0
BUSINESS	C	00	0
BUSINESS	C	OTHERS	0
			6
BUSINESS	M	04	0
BUSINESS	M	06	0
BUSINESS	M	08	0
BUSINESS	M	00	0
BUSINESS	M	OTHERS	0
			0
BUSINESS	W	04	0
BUSINESS	W	06	0
BUSINESS	W	08	0
BUSINESS	W	00	0
BUSINESS	W	OTHERS	0
			0
BUSINESS	OTHERS	04	2
BUSINESS	OTHERS	06	0
BUSINESS	OTHERS	08	0
BUSINESS	OTHERS	00	1
BUSINESS	OTHERS	OTHERS	0
			3
WESTERN1	A	04	0
WESTERN1	A	06	2
WESTERN1	A	08	0
WESTERN1	A	00	0
WESTERN1	A	OTHERS	0
			2
WESTERN1	B	04	0
WESTERN1	B	06	0
WESTERN1	B	08	0

WESTERN1	B	OTHERS	0
			0
WESTERN1	C	04	0
WESTERN1	C	06	0
WESTERN1	C	08	0
WESTERN1	C	00	0
WESTERN1	C	OTHERS	0
			0
WESTERN1	M	04	0
WESTERN1	M	06	0
WESTERN1	M	08	0
WESTERN1	M	00	0
WESTERN1	M	OTHERS	0
			0
WESTERN1	W	04	0
WESTERN1	W	06	0
WESTERN1	W	08	0
WESTERN1	W	00	0
WESTERN1	W	OTHERS	0
			0
WESTERN1	OTHERS	04	0
WESTERN1	OTHERS	06	0
WESTERN1	OTHERS	08	0
WESTERN1	OTHERS	00	0
WESTERN1	OTHERS	OTHERS	0
			0
WESTERN2	A	04	0
WESTERN2	A	06	0
WESTERN2	A	08	0
WESTERN2	A	00	0
WESTERN2	A	OTHERS	0
			0
WESTERN2	B	04	0
WESTERN2	B	06	0
WESTERN2	B	08	0
WESTERN2	B	00	0
WESTERN2	B	OTHERS	0
			0
WESTERN2	C	04	0
WESTERN2	C	06	0
WESTERN2	C	08	0
WESTERN2	C	00	0
WESTERN2	C	OTHERS	0
			0
WESTERN2	M	04	0
WESTERN2	M	06	0
WESTERN2	M	08	0
WESTERN2	M	00	0
WESTERN2	M	OTHERS	0
			0
WESTERN2	W	04	0
WESTERN2	W	06	0
WESTERN2	W	08	0
WESTERN2	W	00	0
WESTERN2	W	OTHERS	0
			0
WESTERN2	OTHERS	04	0
WESTERN2	OTHERS	06	0
WESTERN2	OTHERS	08	0

WESTERN2	OTHERS	OTHERS	0	
				0
CRJE	A	04	48	
CRJE	A	06	13	
CRJE	A	08	43	
CRJE	A	00	0	
CRJE	A	OTHERS	0	
				104
CRJE	B	04	19	
CRJE	B	06	3	
CRJE	B	08	22	
CRJE	B	00	0	
CRJE	B	OTHERS	0	
				44
CRJE	C	04	0	
CRJE	C	06	11	
CRJE	C	08	0	
CRJE	C	00	0	
CRJE	C	OTHERS	0	
				11
CRJE	M	04	0	
CRJE	M	06	4	
CRJE	M	08	0	
CRJE	M	00	0	
CRJE	M	OTHERS	0	
				4
CRJE	W	04	0	
CRJE	W	06	0	
CRJE	W	08	0	
CRJE	W	00	52	
CRJE	W	OTHERS	0	
				52
CRJE	OTHERS	04	0	
CRJE	OTHERS	06	0	
CRJE	OTHERS	08	0	
CRJE	OTHERS	00	4	
CRJE	OTHERS	OTHERS	0	
				4
TSC	A	04	6	
TSC	A	06	14	
TSO	A	08	5	
TSC	A	00	0	
TSO	A	OTHERS	0	
				25
TSC	B	04	0	
TSC	B	06	2	
TSC	B	08	23	
TSC	B	00	0	
TSO	B	OTHERS	0	
				25
TSC	C	04	0	
TSC	C	06	0	
TSC	C	08	0	
TSC	C	00	0	
TSO	C	OTHERS	0	
				0
TSC	M	04	0	
TSC	M	06	0	
TSC	M	08	0	

TSO	M	OTHERS	0	
				0
TSO	W	04	0	
TSO	W	06	0	
TSO	W	08	0	
TSO	W	00	0	
TSO	W	OTHERS	0	
				0
TSO	OTHERS	04	0	
TSO	OTHERS	06	0	
TSO	OTHERS	08	0	
TSO	OTHERS	00	0	
TSO	OTHERS	OTHERS	0	
				0
BATCH	A	04	0	
BATCH	A	06	0	
BATCH	A	08	0	
BATCH	A	00	0	
BATCH	A	OTHERS	0	
				0
BATCH	B	04	0	
BATCH	B	06	0	
BATCH	B	08	0	
BATCH	B	00	0	
BATCH	B	OTHERS	0	
				0
BATCH	C	04	0	
BATCH	C	06	0	
BATCH	C	08	0	
BATCH	C	00	0	
BATCH	C	OTHERS	0	
				0
BATCH	M	04	0	
BATCH	M	06	0	
BATCH	M	08	0	
BATCH	M	00	0	
BATCH	M	OTHERS	0	
				0
BATCH	W	04	0	
BATCH	W	06	0	
BATCH	W	08	0	
BATCH	W	00	981	
BATCH	W	OTHERS	0	
				981
BATCH	OTHERS	04	0	
BATCH	OTHERS	06	0	
BATCH	OTHERS	08	0	
BATCH	OTHERS	00	1	
BATCH	OTHERS	OTHERS	0	
				1
OTHERS	A	04	1	
OTHERS	A	06	13	
OTHERS	A	08	14	
OTHERS	A	00	0	
OTHERS	A	OTHERS	0	
OTHERS	B	04	0	
OTHERS	B	06	1	
OTHERS	B	08	7	
OTHERS	B	00	0	

CTHERS	C	04	7
CTHERS	C	06	29
CTHERS	C	08	9
CTHERS	C	00	0
CTHERS	C	CTHERS	0
CTHERS	M	04	0
CTHERS	M	06	100
CTHERS	M	08	24
CTHERS	M	00	0
CTHERS	M	CTHERS	0
CTHERS	W	04	0
CTHERS	W	06	0
CTHERS	W	08	0
CTHERS	W	00	59
CTHERS	W	CTHERS	0
CTHERS	CTHERS	04	0
CTHERS	OTHERS	06	1
CTHERS	CTHERS	08	0
CTHERS	OTHERS	00	9
CTHERS	OTHERS	OTHERS	0

THE TOTAL NUMBER OF RECORDS CLASSIFIED IS 3957

RTCCDE	PRICR	CLASS	CCCURENCES	SUBTOTALS
CENTRE	04	A	215	
CENTRE	04	B	34	
CENTRE	04	C	10	
CENTRE	04	M	0	
CENTRE	04	W	0	
CENTRE	04	OTHERS	0	
				259
CENTRE	06	A	127	
CENTRE	06	B	27	
CENTRE	06	C	24	
CENTRE	06	M	0	
CENTRE	06	W	0	
CENTRE	06	OTHERS	0	
				178
CENTRE	08	A	41	
CENTRE	08	B	22	
CENTRE	08	C	3	
CENTRE	08	M	0	
CENTRE	08	W	0	
CENTRE	08	OTHERS	0	
				66
CENTRE	00	A	0	
CENTRE	00	B	0	
CENTRE	00	C	0	
CENTRE	00	M	0	
CENTRE	00	W	0	
CENTRE	00	OTHERS	12	
				12
CENTRE	OTHERS	A	0	
CENTRE	OTHERS	B	0	
CENTRE	OTHERS	C	0	
CENTRE	OTHERS	M	0	
CENTRE	OTHERS	W	0	
CENTRE	OTHERS	OTHERS	0	
				0
PHYSICS	04	A	126	
PHYSICS	04	B	9	
PHYSICS	04	C	2	
PHYSICS	04	M	0	
PHYSICS	04	W	0	
PHYSICS	04	OTHERS	1	
				138
PHYSICS	06	A	52	
PHYSICS	06	B	1	
PHYSICS	06	C	1	
PHYSICS	06	M	0	
PHYSICS	06	W	0	
PHYSICS	06	OTHERS	0	
				54
PHYSICS	08	A	1	
PHYSICS	08	B	2	
PHYSICS	08	C	0	
PHYSICS	08	M	0	
PHYSICS	08	W	0	
PHYSICS	08	OTHERS	0	
				3
PHYSICS	00	A	2	

PHYSICS	00	C	0
PHYSICS	00	M	0
PHYSICS	00	W	301
PHYSICS	00	OTHERS	5
			308
PHYSICS	OTHERS	A	0
PHYSICS	OTHERS	B	0
PHYSICS	OTHERS	C	0
PHYSICS	OTHERS	M	0
PHYSICS	OTHERS	W	0
PHYSICS	OTHERS	OTHERS	0
			0
ARTS	04	A	75
ARTS	04	B	9
ARTS	04	C	2
ARTS	04	M	0
ARTS	04	W	0
ARTS	04	OTHERS	1
			87
ARTS	06	A	68
ARTS	06	B	1
ARTS	06	C	0
ARTS	06	M	0
ARTS	06	W	0
ARTS	06	OTHERS	0
			69
ARTS	08	A	8
ARTS	08	B	2
ARTS	08	C	0
ARTS	08	M	0
ARTS	08	W	0
ARTS	08	OTHERS	0
			10
ARTS	00	A	0
ARTS	00	B	0
ARTS	00	C	0
ARTS	00	M	0
ARTS	00	W	1287
ARTS	00	OTHERS	14
			1301
ARTS	OTHERS	A	0
ARTS	OTHERS	B	0
ARTS	OTHERS	C	0
ARTS	OTHERS	M	0
ARTS	OTHERS	W	0
ARTS	OTHERS	OTHERS	0
			0
SCARBCRCU	04	A	0
SCARBCRCU	04	B	4
SCARBCRCU	04	C	1
SCARBCROU	04	M	0
SCARBCRCU	04	W	0
SCARBOROU	04	OTHERS	0
			5
SCARBCRCU	06	A	5
SCARBOROU	06	B	7
SCARBCRCU	06	C	0
SCARBCRCU	06	M	24
SCARBOROU	06	W	0

			36
SCARBOROU	08	A	0
SCARBOROU	08	B	0
SCARBOROU	08	C	0
SCARBOROU	08	M	0
SCARBOROU	08	W	0
SCARBOROU	08	OTHERS	0
			0
SCARBOROU	00	A	0
SCARBOROU	00	B	0
SCARBOROU	00	C	0
SCARBOROU	00	M	0
SCARBOROU	00	W	51
SCARBOROU	00	OTHERS	0
			51
SCARBOROU	OTHERS	A	0
SCARBOROU	OTHERS	B	0
SCARBOROU	OTHERS	C	0
SCARBOROU	OTHERS	M	0
SCARBOROU	OTHERS	W	0
SCARBOROU	OTHERS	OTHERS	0
			0
ERINDALE	04	A	18
ERINDALE	04	B	0
ERINDALE	04	C	1
ERINDALE	04	M	0
ERINDALE	04	W	0
ERINDALE	04	OTHERS	0
			19
ERINDALE	06	A	2
ERINDALE	06	B	0
ERINDALE	06	C	0
ERINDALE	06	M	0
ERINDALE	06	W	0
ERINDALE	06	OTHERS	0
			2
ERINDALE	08	A	0
ERINDALE	08	B	0
ERINDALE	08	C	0
ERINDALE	08	M	0
ERINDALE	08	W	0
ERINDALE	08	OTHERS	0
			0
ERINDALE	00	A	0
ERINDALE	00	B	0
ERINDALE	00	C	0
ERINDALE	00	M	0
ERINDALE	00	W	9
ERINDALE	00	OTHERS	0
			9
ERINDALE	OTHERS	A	0
ERINDALE	OTHERS	B	0
ERINDALE	OTHERS	C	0
ERINDALE	OTHERS	M	0
ERINDALE	OTHERS	W	0
ERINDALE	OTHERS	OTHERS	0
			0
DENTISTRY	04	A	0
DENTISTRY	04	B	0

DENTISTRY 04	M	0
DENTISTRY 04	W	0
DENTISTRY 04	OTHERS	0
		0
DENTISTRY 06	A	0
DENTISTRY 06	B	0
DENTISTRY 06	C	0
DENTISTRY 06	M	0
DENTISTRY 06	W	0
DENTISTRY 06	OTHERS	0
		0
DENTISTRY 08	A	0
DENTISTRY 08	B	0
DENTISTRY 08	C	0
DENTISTRY 08	M	0
DENTISTRY 08	W	0
DENTISTRY 08	OTHERS	0
		0
DENTISTRY 00	A	0
DENTISTRY 00	B	0
DENTISTRY 00	C	0
DENTISTRY 00	M	0
DENTISTRY 00	W	0
DENTISTRY 00	OTHERS	0
		0
DENTISTRY OTHERS	A	0
DENTISTRY OTHERS	B	0
DENTISTRY OTHERS	C	0
DENTISTRY OTHERS	M	0
DENTISTRY OTHERS	W	0
DENTISTRY OTHERS	OTHERS	0
		0
AEROSPACE 04	A	9
AEROSPACE 04	B	0
AEROSPACE 04	C	0
AEROSPACE 04	M	0
AEROSPACE 04	W	0
AEROSPACE 04	OTHERS	0
		9
AEROSPACE 06	A	3
AEROSPACE 06	B	0
AEROSPACE 06	C	0
AEROSPACE 06	M	0
AEROSPACE 06	W	0
AEROSPACE 06	OTHERS	0
		3
AEROSPACE 08	A	0
AEROSPACE 08	B	0
AEROSPACE 08	C	0
AEROSPACE 08	M	0
AEROSPACE 08	W	0
AEROSPACE 08	OTHERS	0
		0
AEROSPACE 00	A	0
AEROSPACE 00	B	0
AEROSPACE 00	C	0
AEROSPACE 00	M	0
AEROSPACE 00	W	51
AEROSPACE 00	OTHERS	0

AEROSPACE	OTHERS	A	0
AEROSPACE	OTHERS	B	0
AEROSPACE	OTHERS	C	0
AEROSPACE	OTHERS	M	0
AEROSPACE	OTHERS	W	0
AEROSPACE	OTHERS	OTHERS	0
			0
BUSINESS	04	A	14
BUSINESS	04	B	0
BUSINESS	04	C	5
BUSINESS	04	M	0
BUSINESS	04	W	0
BUSINESS	04	OTHERS	2
			21
BUSINESS	06	A	10
BUSINESS	06	B	0
BUSINESS	06	C	1
BUSINESS	06	M	0
BUSINESS	06	W	0
BUSINESS	06	OTHERS	0
			11
BUSINESS	08	A	0
BUSINESS	08	B	1
BUSINESS	08	C	0
BUSINESS	08	M	0
BUSINESS	08	W	0
BUSINESS	08	OTHERS	0
			1
BUSINESS	00	A	0
BUSINESS	00	B	0
BUSINESS	00	C	0
BUSINESS	00	M	0
BUSINESS	00	W	0
BUSINESS	00	OTHERS	1
			1
BUSINESS	OTHERS	A	0
BUSINESS	OTHERS	B	0
BUSINESS	OTHERS	C	0
BUSINESS	OTHERS	M	0
BUSINESS	OTHERS	W	0
BUSINESS	OTHERS	OTHERS	0
			0
WESTERN1	04	A	0
WESTERN1	04	B	0
WESTERN1	04	C	0
WESTERN1	04	M	0
WESTERN1	04	W	0
WESTERN1	04	OTHERS	0
			0
WESTERN1	06	A	2
WESTERN1	06	B	0
WESTERN1	06	C	0
WESTERN1	06	M	0
WESTERN1	06	W	0
WESTERN1	06	OTHERS	0
			2
WESTERN1	08	A	0
WESTERN1	08	B	0
WESTERN1	08	C	0

WESTERN1	08	W	0
WESTERN1	08	CTHERS	0
			0
WESTERN1	00	A	0
WESTERN1	00	B	0
WESTERN1	00	C	0
WESTERN1	00	M	0
WESTERN1	00	W	0
WESTERN1	00	OTHERS	0
			0
WESTERN1	OTHERS	A	0
WESTERN1	OTHERS	B	0
WESTERN1	OTHERS	C	0
WESTERN1	OTHERS	M	0
WESTERN1	OTHERS	W	0
WESTERN1	OTHERS	OTHERS	0
			0
WESTERN2	04	A	0
WESTERN2	04	B	0
WESTERN2	04	C	0
WESTERN2	04	M	0
WESTERN2	04	W	0
WESTERN2	04	OTHERS	0
			0
WESTERN2	06	A	0
WESTERN2	06	B	0
WESTERN2	06	C	0
WESTERN2	06	M	0
WESTERN2	06	W	0
WESTERN2	06	OTHERS	0
			0
WESTERN2	08	A	0
WESTERN2	08	B	0
WESTERN2	08	C	0
WESTERN2	08	M	0
WESTERN2	08	W	0
WESTERN2	08	OTHERS	0
			0
WESTERN2	00	A	0
WESTERN2	00	B	0
WESTERN2	00	C	0
WESTERN2	00	M	0
WESTERN2	00	W	0
WESTERN2	00	CTHERS	0
			0
WESTERN2	CTHERS	A	0
WESTERN2	OTHERS	B	0
WESTERN2	CTHERS	C	0
WESTERN2	CTHERS	M	0
WESTERN2	OTHERS	W	0
WESTERN2	OTHERS	OTHERS	0
			0
CRJE	04	A	48
CRJE	04	B	19
CRJE	04	C	0
CRJE	04	M	0
CRJE	04	W	0
CRJE	04	CTHERS	0
			67

CRJE	06	B	3
CRJE	06	C	11
CRJE	06	M	4
CRJE	06	W	0
CRJE	06	OTHERS	0
			31
CRJE	08	A	43
CRJE	08	B	22
CRJE	08	C	0
CRJE	08	M	0
CRJE	08	W	0
CRJE	08	OTHERS	0
			65
CRJE	00	A	0
CRJE	00	B	0
CRJE	00	C	0
CRJE	00	M	0
CRJE	00	W	52
CRJE	00	OTHERS	4
			56
CRJE	OTHERS	A	0
CRJE	OTHERS	B	0
CRJE	OTHERS	C	0
CRJE	OTHERS	M	0
CRJE	OTHERS	W	0
CRJE	OTHERS	OTHERS	0
			0
TSO	04	A	6
TSO	04	B	0
TSO	04	C	0
TSO	04	M	0
TSO	04	W	0
TSO	04	OTHERS	0
			6
TSO	06	A	14
TSO	06	B	2
TSO	06	C	0
TSO	06	M	0
TSO	06	W	0
TSO	06	OTHERS	0
			16
TSO	08	A	5
TSO	08	B	23
TSO	08	C	0
TSO	08	M	0
TSO	08	W	0
TSO	08	OTHERS	0
			28
TSO	00	A	0
TSO	00	B	0
TSO	00	C	0
TSO	00	M	0
TSO	00	W	0
TSO	00	OTHERS	0
			0
TSO	OTHERS	A	0
TSO	OTHERS	B	0
TSO	OTHERS	C	0
TSO	OTHERS	M	0

TSC	CTHERS	CTHERS	0
			0
BATCH	04	A	0
BATCH	04	B	0
BATCH	04	C	0
BATCH	04	M	0
BATCH	04	W	0
BATCH	04	CTHERS	0
			0
BATCH	06	A	0
BATCH	06	B	0
BATCH	06	C	0
BATCH	06	M	0
BATCH	06	W	0
BATCH	06	CTHERS	0
			0
BATCH	08	A	0
BATCH	08	B	0
BATCH	08	C	0
BATCH	08	M	0
BATCH	08	W	0
BATCH	08	CTHERS	0
			0
BATCH	00	A	0
BATCH	00	B	0
BATCH	00	C	0
BATCH	00	M	0
BATCH	00	W	981
BATCH	00	OTHERS	1
			982
BATCH	CTHERS	A	0
BATCH	OTHERS	B	0
BATCH	CTHERS	C	0
BATCH	CTHERS	M	0
BATCH	CTHERS	W	0
BATCH	CTHERS	CTHERS	0
			0
CTHERS	04	A	1
CTHERS	04	B	0
CTHERS	04	C	7
CTHERS	04	M	0
CTHERS	04	W	0
OTHERS	04	CTHERS	0
CTHERS	06	A	13
CTHERS	06	B	1
CTHERS	06	C	29
CTHERS	06	M	100
CTHERS	06	W	0
OTHERS	06	OTHERS	1
CTHERS	08	A	14
CTHERS	08	B	7
CTHERS	08	C	9
CTHERS	08	M	24
OTHERS	08	W	0
CTHERS	08	OTHERS	0
CTHERS	00	A	0
CTHERS	00	B	0
CTHERS	00	C	0
CTHERS	00	M	0

CTHERS	00	CTHERS	9
CTHERS	OTHERS	A	0
CTHERS	CTHERS	B	0
CTHERS	CTHERS	C	0
OTHERS	OTHERS	M	0
CTHERS	CTHERS	W	0
CTHERS	CTHERS	CTHERS	0

THE TOTAL NUMBER OF RECORDS CLASSIFIED IS 3957

PRIOR	RTCODE	CLASS	OCCURENCES	SUBTOTALS
04	CENTRE	A	215	
04	CENTRE	B	34	
04	CENTRE	C	10	
04	CENTRE	M	0	
04	CENTRE	W	0	
04	CENTRE	OTHERS	0	
				259
04	PHYSICS	A	126	
04	PHYSICS	B	9	
04	PHYSICS	C	2	
04	PHYSICS	M	0	
04	PHYSICS	W	0	
04	PHYSICS	OTHERS	1	
				138
04	ARTS	A	75	
04	ARTS	B	9	
04	ARTS	C	2	
04	ARTS	M	0	
04	ARTS	W	0	
04	ARTS	OTHERS	1	
				87
04	SCARBORCU	A	0	
04	SCARBOROU	B	4	
04	SCARBORCU	C	1	
04	SCARBORCU	M	0	
04	SCARBOROU	W	0	
04	SCARBORCU	OTHERS	0	
				5
04	ERINDALE	A	18	
04	ERINDALE	B	0	
04	ERINDALE	C	1	
04	ERINDALE	M	0	
04	ERINDALE	W	0	
04	ERINDALE	OTHERS	0	
				19
04	DENTISTRY	A	0	
04	DENTISTRY	B	0	
04	DENTISTRY	C	0	
04	DENTISTRY	M	0	
04	DENTISTRY	W	0	
04	DENTISTRY	OTHERS	0	
				0
04	AEROSPACE	A	9	
04	AEROSPACE	B	0	
04	AEROSPACE	C	0	
04	AEROSPACE	M	0	
04	AEROSPACE	W	0	
04	AEROSPACE	OTHERS	0	
				9
04	BUSINESS	A	14	
04	BUSINESS	B	0	
04	BUSINESS	C	5	
04	BUSINESS	M	0	
04	BUSINESS	W	0	
04	BUSINESS	OTHERS	2	
				21
04	WESTERN1	A	0	

04	WESTERN1	C	0
04	WESTERN1	M	0
04	WESTERN1	W	0
04	WESTERN1	OTHERS	0
			0
04	WESTERN2	A	0
04	WESTERN2	B	0
04	WESTERN2	C	0
04	WESTERN2	M	0
04	WESTERN2	W	0
04	WESTERN2	OTHERS	0
			0
04	CRJE	A	48
04	CRJE	B	19
04	CRJE	C	0
04	CRJE	M	0
04	CRJE	W	0
04	CRJE	OTHERS	0
			67
04	TSO	A	6
04	TSC	B	0
04	TSC	C	0
04	TSO	M	0
04	TSO	W	0
04	TSC	OTHERS	0
			6
04	BATCH	A	0
04	BATCH	B	0
04	BATCH	C	0
04	BATCH	M	0
04	BATCH	W	0
04	BATCH	OTHERS	0
			0
04	OTHERS	A	1
04	OTHERS	B	0
04	OTHERS	C	7
04	OTHERS	M	0
04	OTHERS	W	0
04	OTHERS	OTHERS	0
			8
06	CENTRE	A	127
06	CENTRE	B	27
06	CENTRE	C	24
06	CENTRE	M	0
06	CENTRE	W	0
06	CENTRE	OTHERS	0
			178
06	PHYSICS	A	52
06	PHYSICS	B	1
06	PHYSICS	C	1
06	PHYSICS	M	0
06	PHYSICS	W	0
06	PHYSICS	OTHERS	0
			54
06	ARTS	A	68
06	ARTS	B	1
06	ARTS	C	0
06	ARTS	M	0
06	ARTS	W	0

				69
06	SCARBORCU	A	5	
06	SCARBORCU	B	7	
06	SCARBORCU	C	0	
06	SCARBORCU	M	24	
06	SCARBORCU	W	0	
06	SCARBORCU	OTHERS	0	
				36
06	ERINDALE	A	2	
06	ERINDALE	B	0	
06	ERINDALE	C	0	
06	ERINDALE	M	0	
06	ERINDALE	W	0	
06	ERINDALE	OTHERS	0	
				2
06	DENTISTRY	A	0	
06	DENTISTRY	B	0	
06	DENTISTRY	C	0	
06	DENTISTRY	M	0	
06	DENTISTRY	W	0	
06	DENTISTRY	OTHERS	0	
				0
06	AEROSPACE	A	3	
06	AEROSPACE	B	0	
06	AEROSPACE	C	0	
06	AEROSPACE	M	0	
06	AEROSPACE	W	0	
06	AEROSPACE	OTHERS	0	
				3
06	BUSINESS	A	10	
06	BUSINESS	B	0	
06	BUSINESS	C	1	
06	BUSINESS	M	0	
06	BUSINESS	W	0	
06	BUSINESS	OTHERS	0	
				11
06	WESTERN1	A	2	
06	WESTERN1	B	0	
06	WESTERN1	C	0	
06	WESTERN1	M	0	
06	WESTERN1	W	0	
06	WESTERN1	OTHERS	0	
				2
06	WESTERN2	A	0	
06	WESTERN2	B	0	
06	WESTERN2	C	0	
06	WESTERN2	M	0	
06	WESTERN2	W	0	
06	WESTERN2	OTHERS	0	
				0
06	CRJE	A	13	
06	CRJE	B	3	
06	CRJE	C	11	
06	CRJE	M	4	
06	CRJE	W	0	
06	CRJE	OTHERS	0	
				31
06	TSO	A	14	
06	TSO	B	2	

06	TSC	M	0
06	TSC	W	0
06	TSC	OTHERS	0
			16
06	BATCH	A	0
06	BATCH	B	0
06	BATCH	C	0
06	BATCH	M	0
06	BATCH	W	0
06	BATCH	OTHERS	0
			0
06	CTHERS	A	13
06	CTHERS	B	1
06	CTHERS	C	29
06	CTHERS	M	100
06	CTHERS	W	0
06	CTHERS	CTHERS	1
			144
08	CENTRE	A	41
08	CENTRE	B	22
08	CENTRE	C	3
08	CENTRE	M	0
08	CENTRE	W	0
08	CENTRE	OTHERS	0
			66
08	PHYSICS	A	1
08	PHYSICS	B	2
08	PHYSICS	C	0
08	PHYSICS	M	0
08	PHYSICS	W	0
08	PHYSICS	CTHERS	0
			3
08	ARTS	A	8
08	ARTS	B	2
08	ARTS	C	0
08	ARTS	M	0
08	ARTS	W	0
08	ARTS	OTHERS	0
			10
08	SCARBORCU	A	0
08	SCARBORCU	B	0
08	SCARBORCU	C	0
08	SCARBORCU	M	0
08	SCARBORCU	W	0
08	SCARBORCU	OTHERS	0
			0
08	ERINDALE	A	0
08	ERINDALE	B	0
08	ERINDALE	C	0
08	ERINDALE	M	0
08	ERINDALE	W	0
08	ERINDALE	CTHERS	0
			0
08	DENTISTRY	A	0
08	DENTISTRY	B	0
08	DENTISTRY	C	0
08	DENTISTRY	M	0
08	DENTISTRY	W	0
08	DENTISTRY	CTHERS	0

08	AEROSPACE	A	0	
08	AEROSPACE	B	0	
08	AEROSPACE	C	0	
08	AEROSPACE	M	0	
08	AEROSPACE	W	0	
08	AEROSPACE	OTHERS	0	
				0
08	BUSINESS	A	0	
08	BUSINESS	B	1	
08	BUSINESS	C	0	
08	BUSINESS	M	0	
08	BUSINESS	W	0	
08	BUSINESS	OTHERS	0	
				1
08	WESTERN1	A	0	
08	WESTERN1	B	0	
08	WESTERN1	C	0	
08	WESTERN1	M	0	
08	WESTERN1	W	0	
08	WESTERN1	OTHERS	0	
				0
08	WESTERN2	A	0	
08	WESTERN2	B	0	
08	WESTERN2	C	0	
08	WESTERN2	M	0	
08	WESTERN2	W	0	
08	WESTERN2	OTHERS	0	
				0
08	CRJE	A	43	
08	CRJE	B	22	
08	CRJE	C	0	
08	CRJE	M	0	
08	CRJE	W	0	
08	CRJE	OTHERS	0	
				65
08	TSO	A	5	
08	TSO	B	23	
08	TSO	C	0	
08	TSO	M	0	
08	TSO	W	0	
08	TSO	OTHERS	0	
				28
08	BATCH	A	0	
08	BATCH	B	0	
08	BATCH	C	0	
08	BATCH	M	0	
08	BATCH	W	0	
08	BATCH	OTHERS	0	
				0
08	OTHERS	A	14	
08	OTHERS	B	7	
08	OTHERS	C	9	
08	OTHERS	M	24	
08	OTHERS	W	0	
08	OTHERS	OTHERS	0	
				54
00	CENTRE	A	0	
00	CENTRE	B	0	
00	CENTRE	C	0	

00	CENTRE	M	0	
00	CENTRE	W	0	
00	CENTRE	OTHERS	12	
				12
00	PHYSICS	A	2	
00	PHYSICS	B	0	
00	PHYSICS	C	0	
00	PHYSICS	M	0	
00	PHYSICS	W	301	
00	PHYSICS	OTHERS	5	
				308
00	ARTS	A	0	
00	ARTS	B	0	
00	ARTS	C	0	
00	ARTS	M	0	
00	ARTS	W	1287	
00	ARTS	OTHERS	14	
				1301
00	SCARBOROU	A	0	
00	SCARBOROU	B	0	
00	SCARBOROU	C	0	
00	SCARBOROU	M	0	
00	SCARBOROU	W	51	
00	SCARBOROU	OTHERS	0	
				51
00	ERINDALE	A	0	
00	ERINDALE	B	0	
00	ERINDALE	C	0	
00	ERINDALE	M	0	
00	ERINDALE	W	9	
00	ERINDALE	OTHERS	0	
				9
00	DENTISTRY	A	0	
00	DENTISTRY	B	0	
00	DENTISTRY	C	0	
00	DENTISTRY	M	0	
00	DENTISTRY	W	0	
00	DENTISTRY	OTHERS	0	
				0
00	AEROSPACE	A	0	
00	AEROSPACE	B	0	
00	AEROSPACE	C	0	
00	AEROSPACE	M	0	
00	AEROSPACE	W	51	
00	AEROSPACE	OTHERS	0	
				51
00	BUSINESS	A	0	
00	BUSINESS	B	0	
00	BUSINESS	C	0	
00	BUSINESS	M	0	
00	BUSINESS	W	0	
00	BUSINESS	OTHERS	1	
				1
00	WESTERN1	A	0	
00	WESTERN1	B	0	
00	WESTERN1	C	0	
00	WESTERN1	M	0	
00	WESTERN1	W	0	
00	WESTERN1	OTHERS	0	
				0

00	WESTERN2	A	0
00	WESTERN2	B	0
00	WESTERN2	C	0
00	WESTERN2	M	0
00	WESTERN2	W	0
00	WESTERN2	OTHERS	0
			0
00	CRJE	A	0
00	CRJE	B	0
00	CRJE	C	0
00	CRJE	M	0
00	CRJE	W	52
00	CRJE	OTHERS	4
			56
00	TSO	A	0
00	TSO	B	0
00	TSO	C	0
00	TSO	M	0
00	TSO	W	0
00	TSO	OTHERS	0
			0
00	BATCH	A	0
00	BATCH	B	0
00	BATCH	C	0
00	BATCH	M	0
00	BATCH	W	981
00	BATCH	OTHERS	1
			982
00	OTHERS	A	0
00	OTHERS	B	0
00	OTHERS	C	0
00	OTHERS	M	0
00	OTHERS	W	59
00	OTHERS	OTHERS	9
			68
OTHERS	CENTRE	A	0
OTHERS	CENTRE	B	0
OTHERS	CENTRE	C	0
OTHERS	CENTRE	M	0
OTHERS	CENTRE	W	0
OTHERS	CENTRE	OTHERS	0
OTHERS	PHYSICS	A	0
OTHERS	PHYSICS	B	0
OTHERS	PHYSICS	C	0
OTHERS	PHYSICS	M	0
OTHERS	PHYSICS	W	0
OTHERS	PHYSICS	OTHERS	0
OTHERS	ARTS	A	0
OTHERS	ARTS	B	0
OTHERS	ARTS	C	0
OTHERS	ARTS	M	0
OTHERS	ARTS	W	0
OTHERS	ARTS	OTHERS	0
OTHERS	SCARBOROU	A	0
OTHERS	SCARBOROU	B	0
OTHERS	SCARBOROU	C	0
OTHERS	SCARBOROU	M	0
OTHERS	SCARBOROU	W	0
OTHERS	SCARBOROU	OTHERS	0
OTHERS	ERINDALE	A	0

OTHERS	ERINDALE	B	0
OTHERS	ERINDALE	C	0
OTHERS	ERINDALE	M	0
OTHERS	ERINDALE	W	0
OTHERS	ERINDALE	OTHERS	0
OTHERS	DENTISTRY	A	0
OTHERS	DENTISTRY	B	0
OTHERS	DENTISTRY	C	0
OTHERS	DENTISTRY	M	0
OTHERS	DENTISTRY	W	0
OTHERS	DENTISTRY	OTHERS	0
OTHERS	AEROSPACE	A	0
OTHERS	AEROSPACE	B	0
OTHERS	AEROSPACE	C	0
OTHERS	AEROSPACE	M	0
OTHERS	AEROSPACE	W	0
OTHERS	AEROSPACE	OTHERS	0
OTHERS	BUSINESS	A	0
OTHERS	BUSINESS	B	0
OTHERS	BUSINESS	C	0
OTHERS	BUSINESS	M	0
OTHERS	BUSINESS	W	0
OTHERS	BUSINESS	OTHERS	0
OTHERS	WESTERN1	A	0
OTHERS	WESTERN1	B	0
OTHERS	WESTERN1	C	0
OTHERS	WESTERN1	M	0
OTHERS	WESTERN1	W	0
OTHERS	WESTERN1	OTHERS	0
OTHERS	WESTERN2	A	0
OTHERS	WESTERN2	B	0
OTHERS	WESTERN2	C	0
OTHERS	WESTERN2	M	0
OTHERS	WESTERN2	W	0
OTHERS	WESTERN2	OTHERS	0
OTHERS	CRJE	A	0
OTHERS	CRJE	B	0
OTHERS	CRJE	C	0
OTHERS	CRJE	M	0
OTHERS	CRJE	W	0
OTHERS	CRJE	OTHERS	0
OTHERS	TSO	A	0
OTHERS	TSO	B	0
OTHERS	TSO	C	0
OTHERS	TSO	M	0
OTHERS	TSO	W	0
OTHERS	TSO	OTHERS	0
OTHERS	BATCH	A	0
OTHERS	BATCH	B	0
OTHERS	BATCH	C	0
OTHERS	BATCH	M	0
OTHERS	BATCH	W	0
OTHERS	BATCH	OTHERS	0
OTHERS	OTHERS	A	0
OTHERS	OTHERS	B	0
OTHERS	OTHERS	C	0
OTHERS	OTHERS	M	0
OTHERS	OTHERS	W	0
OTHERS	OTHERS	OTHERS	0

THE TOTAL NUMBER OF RECORDS CLASSIFIED IS 4231

CLASSIFY/ORDER THIS DATA IS GENERATED BY LINE 31

PRIOR	CLASS	RTCODE	OCCURENCES	SUBTOTALS
04	A	CENTRE	215	
04	A	PHYSICS	126	
04	A	ARTS	75	
04	A	SCARBOROU	0	
04	A	ERINDALE	18	
04	A	DENTISTRY	0	
04	A	AEROSPACE	9	
04	A	BUSINESS	14	
04	A	WESTERN1	0	
04	A	WESTERN2	0	
04	A	CRJE	48	
04	A	TSO	6	
04	A	BATCH	0	
04	A	OTHERS	1	
				512
04	B	CENTRE	34	
04	B	PHYSICS	9	
04	B	ARTS	9	
04	B	SCARBOROU	4	
04	B	ERINDALE	0	
04	B	DENTISTRY	0	
04	B	AEROSPACE	0	
04	B	BUSINESS	0	
04	B	WESTERN1	0	
04	B	WESTERN2	0	
04	B	CRJE	19	
04	B	TSO	0	
04	B	BATCH	0	
04	B	OTHERS	0	
				75
04	C	CENTRE	10	
04	C	PHYSICS	2	
04	C	ARTS	2	
04	C	SCARBOROU	1	
04	C	ERINDALE	1	
04	C	DENTISTRY	0	
04	C	AEROSPACE	0	
04	C	BUSINESS	5	
04	C	WESTERN1	0	
04	C	WESTERN2	0	
04	C	CRJE	0	
04	C	TSO	0	
04	C	BATCH	0	
04	C	OTHERS	7	
				28
04	M	CENTRE	0	
04	M	PHYSICS	0	
04	M	ARTS	0	
04	M	SCARBOROU	0	
04	M	ERINDALE	0	
04	M	DENTISTRY	0	
04	M	AEROSPACE	0	
04	M	BUSINESS	0	
04	M	WESTERN1	0	
04	M	WESTERN2	0	
04	M	CRJE	0	
04	M	TSO	0	

04	M	BATCH	0	
04	M	OTHERS	0	
				0
04	W	CENTRE	0	
04	W	PHYSICS	0	
04	W	ARTS	0	
04	W	SCARBORDU	0	
04	W	ERINDALE	0	
04	W	DENTISTRY	0	
04	W	AEROSPACE	0	
04	W	BUSINESS	0	
04	W	WESTERN1	0	
04	W	WESTERN2	0	
04	W	CRJE	0	
04	W	TSO	0	
04	W	BATCH	0	
04	W	OTHERS	0	
				0
04	OTHERS	CENTRE	0	
04	OTHERS	PHYSICS	1	
04	OTHERS	ARTS	1	
04	OTHERS	SCARBORDU	0	
04	OTHERS	ERINDALE	0	
04	OTHERS	DENTISTRY	0	
04	OTHERS	AEROSPACE	0	
04	OTHERS	BUSINESS	2	
04	OTHERS	WESTERN1	0	
04	OTHERS	WESTERN2	0	
04	OTHERS	CRJE	0	
04	OTHERS	TSO	0	
04	OTHERS	BATCH	0	
04	OTHERS	OTHERS	0	
				4
06	A	CENTRE	127	
06	A	PHYSICS	52	
06	A	ARTS	68	
06	A	SCARBORDU	5	
06	A	ERINDALE	2	
06	A	DENTISTRY	0	
06	A	AEROSPACE	3	
06	A	BUSINESS	10	
06	A	WESTERN1	2	
06	A	WESTERN2	0	
06	A	CRJE	13	
06	A	TSO	14	
06	A	BATCH	0	
06	A	OTHERS	13	
				309
06	B	CENTRE	27	
06	B	PHYSICS	1	
06	B	ARTS	1	
06	B	SCARBORDU	7	
06	B	ERINDALE	0	
06	B	DENTISTRY	0	
06	B	AEROSPACE	0	
06	B	BUSINESS	0	
06	B	WESTERN1	0	
06	B	WESTERN2	0	
06	B	CRJE	3	
06	B	TSO	2	

06	B	BATCH	0	
06	B	OTHERS	1	
				42
06	C	CENTRE	24	
06	C	PHYSICS	1	
06	C	ARTS	0	
06	C	SCARBORDU	0	
06	C	ERINDALE	0	
06	C	DENTISTRY	0	
06	C	AEROSPACE	0	
06	C	BUSINESS	1	
06	C	WESTERN1	0	
06	C	WESTERN2	0	
06	C	CRJE	11	
06	C	TSO	0	
06	C	BATCH	0	
06	C	OTHERS	29	
				66
06	M	CENTRE	0	
06	M	PHYSICS	0	
06	M	ARTS	0	
06	M	SCARBORDU	24	
06	M	ERINDALE	0	
06	M	DENTISTRY	0	
06	M	AEROSPACE	0	
06	M	BUSINESS	0	
06	M	WESTERN1	0	
06	M	WESTERN2	0	
06	M	CRJE	4	
06	M	TSO	0	
06	M	BATCH	0	
06	M	OTHERS	100	
				128
06	W	CENTRE	0	
06	W	PHYSICS	0	
06	W	ARTS	0	
06	W	SCARBOPOU	0	
06	W	ERINDALE	0	
06	W	DENTISTRY	0	
06	W	AEROSPACE	0	
06	W	BUSINESS	0	
06	W	WESTERN1	0	
06	W	WESTERN2	0	
06	W	CRJE	0	
06	W	TSO	0	
06	W	BATCH	0	
06	W	OTHERS	0	
				0
06	OTHERS	CENTRE	0	
06	OTHERS	PHYSICS	0	
06	OTHERS	ARTS	0	
06	OTHERS	SCARBORDU	0	
06	OTHERS	ERINDALE	0	
06	OTHERS	DENTISTRY	0	
06	OTHERS	AEROSPACE	0	
06	OTHERS	BUSINESS	0	
06	OTHERS	WESTERN1	0	
06	OTHERS	WESTERN2	0	
06	OTHERS	CRJE	0	
06	OTHERS	TSO	0	

08	OTHERS	BATCH	0
06	OTHERS	OTHERS	1
			1
08	A	CENTRE	41
08	A	PHYSICS	1
08	A	ARTS	8
08	A	SCARBORDU	0
08	A	ERINDALE	0
08	A	DENTISTRY	0
08	A	AEROSPACE	0
08	A	BUSINESS	0
08	A	WESTERN1	0
08	A	WESTERN2	0
08	A	CRJE	43
08	A	TSO	5
08	A	BATCH	0
08	A	OTHERS	14
			112
08	B	CENTRE	22
08	B	PHYSICS	2
08	B	ARTS	2
08	B	SCARBORDU	0
08	B	ERINDALE	0
08	B	DENTISTRY	0
08	B	AEROSPACE	0
08	B	BUSINESS	1
08	B	WESTERN1	0
08	B	WESTERN2	0
08	B	CRJE	22
08	B	TSO	23
08	B	BATCH	0
08	B	OTHERS	7
			79
08	C	CENTRE	3
08	C	PHYSICS	0
08	C	ARTS	0
08	C	SCARBORDU	0
08	C	ERINDALE	0
08	C	DENTISTRY	0
08	C	AEROSPACE	0
08	C	BUSINESS	0
08	C	WESTERN1	0
08	C	WESTERN2	0
08	C	CRJE	0
08	C	TSO	0
08	C	BATCH	0
08	C	OTHERS	9
			12
08	M	CENTRE	0
08	M	PHYSICS	0
08	M	ARTS	0
08	M	SCARBORDU	0
08	M	ERINDALE	0
08	M	DENTISTRY	0
08	M	AEROSPACE	0
08	M	BUSINESS	0
08	M	WESTERN1	0
08	M	WESTERN2	0
08	M	CRJE	0
08	M	TSO	0

08	M	BATCH	0
08	M	OTHERS	24
			24
08	W	CENTRE	0
08	W	PHYSICS	0
08	W	ARTS	0
08	W	SCARBORDU	0
08	W	ERINDALE	0
08	W	DENTISTRY	0
08	W	AEROSPACE	0
08	W	BUSINESS	0
08	W	WESTERN1	0
08	W	WESTERN2	0
08	W	CRJE	0
08	W	TSO	0
08	W	BATCH	0
08	W	OTHERS	0
			0
08	OTHERS	CENTRE	0
08	OTHERS	PHYSICS	0
08	OTHERS	ARTS	0
08	OTHERS	SCARBORDU	0
08	OTHERS	ERINDALE	0
08	OTHERS	DENTISTRY	0
08	OTHERS	AEROSPACE	0
08	OTHERS	BUSINESS	0
08	OTHERS	WESTERN1	0
08	OTHERS	WESTERN2	0
08	OTHERS	CRJE	0
08	OTHERS	TSO	0
08	OTHERS	BATCH	0
08	OTHERS	OTHERS	0
			0
00	A	CENTRE	0
00	A	PHYSICS	2
00	A	ARTS	0
00	A	SCARBORDU	0
00	A	ERINDALE	0
00	A	DENTISTRY	0
00	A	AEROSPACE	0
00	A	BUSINESS	0
00	A	WESTERN1	0
00	A	WESTERN2	0
00	A	CRJE	0
00	A	TSO	0
00	A	BATCH	0
00	A	OTHERS	0
			2
00	B	CENTRE	0
00	B	PHYSICS	0
00	B	ARTS	0
00	B	SCARBORDU	0
00	B	ERINDALE	0
00	B	DENTISTRY	0
00	B	AEROSPACE	0
00	B	BUSINESS	0
00	B	WESTERN1	0
00	B	WESTERN2	0
00	B	CRJE	0
00	B	TSO	0

00	B	BATCH	0	
00	B	OTHERS	0	
				0
00	C	CENTRE	0	
00	C	PHYSICS	0	
00	C	ARTS	0	
00	C	SCARBORDU	0	
00	C	ERINDALE	0	
00	C	DENTISTRY	0	
00	C	AEROSPACE	0	
00	C	BUSINESS	0	
00	C	WESTERN1	0	
00	C	WESTERN2	0	
00	C	CPJE	0	
00	C	TSO	0	
00	C	BATCH	0	
00	C	OTHERS	0	
				0
00	M	CENTRE	0	
00	M	PHYSICS	0	
00	M	ARTS	0	
00	M	SCARBORDU	0	
00	M	ERINDALE	0	
00	M	DENTISTRY	0	
00	M	AEROSPACE	0	
00	M	BUSINESS	0	
00	M	WESTERN1	0	
00	M	WESTERN2	0	
00	M	CRJE	0	
00	M	TSO	0	
00	M	BATCH	0	
00	M	OTHERS	0	
				0
00	W	CENTRE	0	
00	W	PHYSICS	301	
00	W	ARTS	1287	
00	W	SCARBORDU	51	
00	W	ERINDALE	9	
00	W	DENTISTRY	0	
00	W	AEROSPACE	51	
00	W	BUSINESS	0	
00	W	WESTERN1	0	
00	W	WESTERN2	0	
00	W	CRJE	52	
00	W	TSO	0	
00	W	BATCH	981	
00	W	OTHERS	59	
				2791
00	OTHERS	CENTRE	12	
00	OTHERS	PHYSICS	5	
00	OTHERS	ARTS	14	
00	OTHERS	SCARBORDU	0	
00	OTHERS	ERINDALE	0	
00	OTHERS	DENTISTRY	0	
00	OTHERS	AEROSPACE	0	
00	OTHERS	BUSINESS	1	
00	OTHERS	WESTERN1	0	
00	OTHERS	WESTERN2	0	
00	OTHERS	CRJE	4	
00	OTHERS	TSO	0	

00	OTHERS	BATCH	1
00	OTHERS	OTHERS	9
OTHERS	A	CENTRE	0
OTHERS	A	PHYSICS	0
OTHERS	A	ARTS	0
OTHERS	A	SCARBORO	0
OTHERS	A	ERINDALE	0
OTHERS	A	DENTISTRY	0
OTHERS	A	AEROSPACE	0
OTHERS	A	BUSINESS	0
OTHERS	A	WESTERN1	0
OTHERS	A	WESTERN2	0
OTHERS	A	CRJE	0
OTHERS	A	TSO	0
OTHERS	A	BATCH	0
OTHERS	A	OTHERS	0
OTHERS	B	CENTRE	0
OTHERS	B	PHYSICS	0
OTHERS	B	ARTS	0
OTHERS	B	SCARBORO	0
OTHERS	B	ERINDALE	0
OTHERS	B	DENTISTRY	0
OTHERS	B	AEROSPACE	0
OTHERS	B	BUSINESS	0
OTHERS	B	WESTERN1	0
OTHERS	B	WESTERN2	0
OTHERS	B	CRJE	0
OTHERS	B	TSO	0
OTHERS	B	BATCH	0
OTHERS	B	OTHERS	0
OTHERS	C	CENTRE	0
OTHERS	C	PHYSICS	0
OTHERS	C	ARTS	0
OTHERS	C	SCARBORO	0
OTHERS	C	ERINDALE	0
OTHERS	C	DENTISTRY	0
OTHERS	C	AEROSPACE	0
OTHERS	C	BUSINESS	0
OTHERS	C	WESTERN1	0
OTHERS	C	WESTERN2	0
OTHERS	C	CRJE	0
OTHERS	C	TSO	0
OTHERS	C	BATCH	0
OTHERS	C	OTHERS	0
OTHERS	M	CENTRE	0
OTHERS	M	PHYSICS	0
OTHERS	M	ARTS	0
OTHERS	M	SCARBORO	0
OTHERS	M	ERINDALE	0
OTHERS	M	DENTISTRY	0
OTHERS	M	AEROSPACE	0
OTHERS	M	BUSINESS	0
OTHERS	M	WESTERN1	0
OTHERS	M	WESTERN2	0
OTHERS	M	CRJE	0
OTHERS	M	TSO	0
OTHERS	M	BATCH	0
OTHERS	M	OTHERS	0
OTHERS	W	CENTRE	0

OTHERS	W	PHYSICS	0
OTHERS	W	ARTS	0
OTHERS	W	SCARBOROU	0
OTHERS	W	ERINDALE	0
OTHERS	W	DENTISTRY	0
OTHERS	W	AEROSPACE	0
OTHERS	W	BUSINESS	0
OTHERS	W	WESTERN1	0
OTHERS	W	WESTERN2	0
OTHERS	W	CRJE	0
OTHERS	W	TSO	0
OTHERS	W	BATCH	0
OTHERS	W	OTHERS	0
OTHERS	OTHERS	CENTRE	0
OTHERS	OTHERS	PHYSICS	0
OTHERS	OTHERS	ARTS	0
OTHERS	OTHERS	SCARBOROU	0
OTHERS	OTHERS	ERINDALE	0
OTHERS	OTHERS	DENTISTRY	0
OTHERS	OTHERS	AEROSPACE	0
OTHERS	OTHERS	BUSINESS	0
OTHERS	OTHERS	WESTERN1	0
OTHERS	OTHERS	WESTERN2	0
OTHERS	OTHERS	CRJE	0
OTHERS	OTHERS	TSO	0
OTHERS	OTHERS	BATCH	0
OTHERS	OTHERS	OTHERS	0

THE TOTAL NUMBER OF RECORDS CLASSIFIED IS 4231

CLASSIFY/ORDER THIS DATA IS GENERATED BY LINE 32

CLASS	RTCODE	PRIOR	OCCURENCES	SUBTOTALS
A	CENTRE	04	215	
A	CENTRE	06	127	
A	CENTRE	08	41	
A	CENTRE	00	0	
A	CENTRE	OTHERS	0	
				383
A	PHYSICS	04	126	
A	PHYSICS	06	52	
A	PHYSICS	08	1	
A	PHYSICS	00	2	
A	PHYSICS	OTHERS	0	
				181
A	ARTS	04	75	
A	ARTS	06	68	
A	ARTS	08	8	
A	ARTS	00	0	
A	ARTS	OTHERS	0	
				151
A	SCARBORDU	04	0	
A	SCARBORDU	06	5	
A	SCARBORDU	08	0	
A	SCARBORDU	00	0	
A	SCARBORDU	OTHERS	0	
				5
A	ERINDALE	04	18	
A	ERINDALE	06	2	
A	ERINDALE	08	0	
A	ERINDALE	00	0	
A	ERINDALE	OTHERS	0	
				20
A	DENTISTRY	04	0	
A	DENTISTRY	06	0	
A	DENTISTRY	08	0	
A	DENTISTRY	00	0	
A	DENTISTRY	OTHERS	0	
				0
A	AEROSPACE	04	9	
A	AEROSPACE	06	3	
A	AEROSPACE	08	0	
A	AEROSPACE	00	0	
A	AEROSPACE	OTHERS	0	
				12
A	BUSINESS	04	14	
A	BUSINESS	06	10	
A	BUSINESS	08	0	
A	BUSINESS	00	0	
A	BUSINESS	OTHERS	0	
				24
A	WESTERN1	04	0	
A	WESTERN1	06	2	
A	WESTERN1	08	0	
A	WESTERN1	00	0	
A	WESTERN1	OTHERS	0	
				2
A	WESTERN2	04	0	
A	WESTERN2	06	0	
A	WESTERN2	08	0	

A	WESTERN2	00	0	
A	WESTERN2	OTHERS	0	
				0
A	CRJE	04	48	
A	CRJE	06	13	
A	CRJE	08	43	
A	CRJE	00	0	
A	CRJE	OTHERS	0	
				104
A	TSO	04	6	
A	TSO	06	14	
A	TSO	08	5	
A	TSO	00	0	
A	TSO	OTHERS	0	
				25
A	BATCH	04	0	
A	BATCH	06	0	
A	BATCH	08	0	
A	BATCH	00	0	
A	BATCH	OTHERS	0	
				0
A	OTHERS	04	1	
A	OTHERS	06	13	
A	OTHERS	08	14	
A	OTHERS	00	0	
A	OTHERS	OTHERS	0	
				28
B	CENTRE	04	34	
B	CENTRE	06	27	
B	CENTRE	08	22	
B	CENTRE	00	0	
B	CENTRE	OTHERS	0	
				83
B	PHYSICS	04	9	
B	PHYSICS	06	1	
B	PHYSICS	08	2	
B	PHYSICS	00	0	
B	PHYSICS	OTHERS	0	
				12
B	ARTS	04	9	
B	ARTS	06	1	
B	ARTS	08	2	
B	ARTS	00	0	
B	ARTS	OTHERS	0	
				12
B	SCARBORDU	04	4	
B	SCARBORDU	06	7	
B	SCARBORDU	08	0	
B	SCARBORDU	00	0	
B	SCARBORDU	OTHERS	0	
				11
B	ERINDALE	04	0	
B	ERINDALE	06	0	
B	ERINDALE	08	0	
B	ERINDALE	00	0	
B	ERINDALE	OTHERS	0	
				0
B	DENTISTRY	04	0	
B	DENTISTRY	06	0	
B	DENTISTRY	08	0	

B	DENTISTRY	00	0	
B	DENTISTRY	OTHERS	0	
				0
B	AEROSPACE	04	0	
B	AEROSPACE	06	0	
B	AEROSPACE	08	0	
B	AEROSPACE	00	0	
B	AEROSPACE	OTHERS	0	
				0
B	BUSINESS	04	0	
B	BUSINESS	06	0	
B	BUSINESS	08	1	
B	BUSINESS	00	0	
B	BUSINESS	OTHERS	0	
				1
B	WESTERN1	04	0	
B	WESTERN1	06	0	
B	WESTERN1	08	0	
B	WESTERN1	00	0	
B	WESTERN1	OTHERS	0	
				0
B	WESTERN2	04	0	
B	WESTERN2	06	0	
B	WESTERN2	08	0	
B	WESTERN2	00	0	
B	WESTERN2	OTHERS	0	
				0
B	CRJE	04	19	
B	CRJE	06	3	
B	CRJE	08	22	
B	CRJE	00	0	
B	CRJE	OTHERS	0	
				44
B	TSO	04	0	
B	TSO	06	2	
B	TSO	08	23	
B	TSO	00	0	
B	TSO	OTHERS	0	
				25
B	BATCH	04	0	
B	BATCH	06	0	
B	BATCH	08	0	
B	BATCH	00	0	
B	BATCH	OTHERS	0	
				0
B	OTHERS	04	0	
B	OTHERS	06	1	
B	OTHERS	08	7	
B	OTHERS	00	0	
B	OTHERS	OTHERS	0	
				8
C	CENTRE	04	10	
C	CENTRE	06	24	
C	CENTRE	08	3	
C	CENTRE	00	0	
C	CENTRE	OTHERS	0	
				37
C	PHYSICS	04	2	
C	PHYSICS	06	1	
C	PHYSICS	08	0	

C	PHYSICS	00	0	
C	PHYSICS	OTHERS	0	
				3
C	ARTS	04	2	
C	ARTS	06	0	
C	ARTS	08	0	
C	ARTS	00	0	
C	ARTS	OTHERS	0	
				2
C	SCARBOROU	04	1	
C	SCARBOROU	06	0	
C	SCARBOROU	08	0	
C	SCARBOROU	00	0	
C	SCARBOROU	OTHERS	0	
				1
C	ERINDALE	04	1	
C	ERINDALE	06	0	
C	ERINDALE	08	0	
C	ERINDALE	00	0	
C	ERINDALE	OTHERS	0	
				1
C	DENTISTRY	04	0	
C	DENTISTRY	06	0	
C	DENTISTRY	08	0	
C	DENTISTRY	00	0	
C	DENTISTRY	OTHERS	0	
				0
C	AEROSPACE	04	0	
C	AEROSPACE	06	0	
C	AEROSPACE	08	0	
C	AEROSPACE	00	0	
C	AEROSPACE	OTHERS	0	
				0
C	BUSINESS	04	5	
C	BUSINESS	06	1	
C	BUSINESS	08	0	
C	BUSINESS	00	0	
C	BUSINESS	OTHERS	0	
				6
C	WESTERN1	04	0	
C	WESTERN1	06	0	
C	WESTERN1	08	0	
C	WESTERN1	00	0	
C	WESTERN1	OTHERS	0	
				0
C	WESTERN2	04	0	
C	WESTERN2	06	0	
C	WESTERN2	08	0	
C	WESTERN2	00	0	
C	WESTERN2	OTHERS	0	
				0
C	CRJE	04	0	
C	CRJE	06	11	
C	CRJE	08	0	
C	CRJE	00	0	
C	CRJE	OTHERS	0	
				11
C	TSO	04	0	
C	TSO	06	0	
C	TSO	08	0	

C	TSO	00	0	
C	TSO	OTHERS	0	
				0
C	BATCH	04	0	
C	BATCH	06	0	
C	BATCH	08	0	
C	BATCH	00	0	
C	BATCH	OTHERS	0	
				0
C	OTHERS	04	7	
C	OTHERS	06	29	
C	OTHERS	08	9	
C	OTHERS	00	0	
C	OTHERS	OTHERS	0	
				45
M	CENTRE	04	0	
M	CENTRE	06	0	
M	CENTRE	08	0	
M	CENTRE	00	0	
M	CENTRE	OTHERS	0	
				0
M	PHYSICS	04	0	
M	PHYSICS	06	0	
M	PHYSICS	08	0	
M	PHYSICS	00	0	
M	PHYSICS	OTHERS	0	
				0
M	ARTS	04	0	
M	ARTS	06	0	
M	ARTS	08	0	
M	ARTS	00	0	
M	ARTS	OTHERS	0	
				0
M	SCARBOROU	04	0	
M	SCARBOROU	06	24	
M	SCARBOROU	08	0	
M	SCARBOROU	00	0	
M	SCARBOROU	OTHERS	0	
				24
M	ERINDALE	04	0	
M	ERINDALE	06	0	
M	ERINDALE	08	0	
M	ERINDALE	00	0	
M	ERINDALE	OTHERS	0	
				0
M	DENTISTRY	04	0	
M	DENTISTRY	06	0	
M	DENTISTRY	08	0	
M	DENTISTRY	00	0	
M	DENTISTRY	OTHERS	0	
				0
M	AEROSPACE	04	0	
M	AEROSPACE	06	0	
M	AEROSPACE	08	0	
M	AEROSPACE	00	0	
M	AEROSPACE	OTHERS	0	
				0
M	BUSINESS	04	0	
M	BUSINESS	06	0	
M	BUSINESS	08	0	

M	BUSINESS	00	0	
M	BUSINESS	OTHERS	0	
				0
M	WESTERN1	04	0	
M	WESTERN1	06	0	
M	WESTERN1	08	0	
M	WESTERN1	00	0	
M	WESTERN1	OTHERS	0	
				0
M	WESTERN2	04	0	
M	WESTERN2	06	0	
M	WESTERN2	08	0	
M	WESTERN2	00	0	
M	WESTERN2	OTHERS	0	
				0
M	CRJE	04	0	
M	CRJE	06	4	
M	CRJE	08	0	
M	CRJE	00	0	
M	CRJE	OTHERS	0	
				4
M	TSO	04	0	
M	TSO	06	0	
M	TSO	08	0	
M	TSO	00	0	
M	TSO	OTHERS	0	
				0
M	BATCH	04	0	
M	BATCH	06	0	
M	BATCH	08	0	
M	BATCH	00	0	
M	BATCH	OTHERS	0	
				0
M	OTHERS	04	0	
M	OTHERS	06	100	
M	OTHERS	08	24	
M	OTHERS	00	0	
M	OTHERS	OTHERS	0	
				124
W	CENTRE	04	0	
W	CENTRE	06	0	
W	CENTRE	08	0	
W	CENTRE	00	0	
W	CENTRE	OTHERS	0	
				0
W	PHYSICS	04	0	
W	PHYSICS	06	0	
W	PHYSICS	08	0	
W	PHYSICS	00	301	
W	PHYSICS	OTHERS	0	
				301
W	ARTS	04	0	
W	ARTS	06	0	
W	ARTS	08	0	
W	ARTS	00	1287	
W	ARTS	OTHERS	0	
				1287
W	SCARBORDU	04	0	
W	SCARBORDU	06	0	
W	SCARBORDU	08	0	

W	SCARBOROU	00	51	
W	SCARBOROU	OTHERS	0	
				51
W	ERINDALE	04	0	
W	ERINDALE	06	0	
W	ERINDALE	08	0	
W	ERINDALE	00	9	
W	ERINDALE	OTHERS	0	
				9
W	DENTISTRY	04	0	
W	DENTISTRY	06	0	
W	DENTISTRY	08	0	
W	DENTISTRY	00	0	
W	DENTISTRY	OTHERS	0	
				0
W	AEROSPACE	04	0	
W	AEROSPACE	06	0	
W	AEROSPACE	08	0	
W	AEROSPACE	00	51	
W	AEROSPACE	OTHERS	0	
				51
W	BUSINESS	04	0	
W	BUSINESS	06	0	
W	BUSINESS	08	0	
W	BUSINESS	00	0	
W	BUSINESS	OTHERS	0	
				0
W	WESTERN1	04	0	
W	WESTERN1	06	0	
W	WESTERN1	08	0	
W	WESTERN1	00	0	
W	WESTERN1	OTHERS	0	
				0
W	WESTERN2	04	0	
W	WESTERN2	06	0	
W	WESTERN2	08	0	
W	WESTERN2	00	0	
W	WESTERN2	OTHERS	0	
				0
W	CRJE	04	0	
W	CRJE	06	0	
W	CRJE	08	0	
W	CRJE	00	52	
W	CRJE	OTHERS	0	
				52
W	TSO	04	0	
W	TSO	06	0	
W	TSO	08	0	
W	TSO	00	0	
W	TSO	OTHERS	0	
				0
W	BATCH	04	0	
W	BATCH	06	0	
W	BATCH	08	0	
W	BATCH	00	981	
W	BATCH	OTHERS	0	
				981
W	OTHERS	04	0	
W	OTHERS	06	0	
W	OTHERS	08	0	

W	OTHERS	00	59
W	OTHERS	OTHERS	0
OTHERS	CENTRE	04	0
OTHERS	CENTRE	06	0
OTHERS	CENTRE	08	0
OTHERS	CENTRE	00	12
OTHERS	CENTRE	OTHERS	0
OTHERS	PHYSICS	04	1
OTHERS	PHYSICS	06	0
OTHERS	PHYSICS	08	0
OTHERS	PHYSICS	00	5
OTHERS	PHYSICS	OTHERS	0
OTHERS	ARTS	04	1
OTHERS	ARTS	06	0
OTHERS	ARTS	08	0
OTHERS	ARTS	00	14
OTHERS	ARTS	OTHERS	0
OTHERS	SCARBOROU	04	0
OTHERS	SCARBOROU	06	0
OTHERS	SCARBOROU	08	0
OTHERS	SCARBOROU	00	0
OTHERS	SCARBOROU	OTHERS	0
OTHERS	ERINDALE	04	0
OTHERS	ERINDALE	06	0
OTHERS	ERINDALE	08	0
OTHERS	ERINDALE	00	0
OTHERS	ERINDALE	OTHERS	0
OTHERS	DENTISTRY	04	0
OTHERS	DENTISTRY	06	0
OTHERS	DENTISTRY	08	0
OTHERS	DENTISTRY	00	0
OTHERS	DENTISTRY	OTHERS	0
OTHERS	AEROSPACE	04	0
OTHERS	AEROSPACE	06	0
OTHERS	AEROSPACE	08	0
OTHERS	AEROSPACE	00	0
OTHERS	AEROSPACE	OTHERS	0
OTHERS	BUSINESS	04	2
OTHERS	BUSINESS	06	0
OTHERS	BUSINESS	08	0
OTHERS	BUSINESS	00	1
OTHERS	BUSINESS	OTHERS	0
OTHERS	WESTERN1	04	0
OTHERS	WESTERN1	06	0
OTHERS	WESTERN1	08	0
OTHERS	WESTERN1	00	0
OTHERS	WESTERN1	OTHERS	0
OTHERS	WESTERN2	04	0
OTHERS	WESTERN2	06	0
OTHERS	WESTERN2	08	0
OTHERS	WESTERN2	00	0
OTHERS	WESTERN2	OTHERS	0
OTHERS	CRJE	04	0
OTHERS	CRJE	06	0
OTHERS	CRJE	08	0
OTHERS	CRJE	00	4
OTHERS	CRJE	OTHERS	0
OTHERS	TSO	04	0
OTHERS	TSO	06	0

OTHERS	TSO	08	0
OTHERS	TSO	00	0
OTHERS	TSO	OTHERS	0
OTHERS	BATCH	04	0
OTHERS	BATCH	06	0
OTHERS	BATCH	08	0
OTHERS	BATCH	00	1
OTHERS	BATCH	OTHERS	0
OTHERS	OTHERS	04	0
OTHERS	OTHERS	06	1
OTHERS	OTHERS	08	0
OTHERS	OTHERS	00	9
OTHERS	OTHERS	OTHERS	0

THE TOTAL NUMBER OF RECORDS CLASSIFIED IS 4180

CLASSIFY/ORDER THIS DATA IS GENERATED BY LINE 33

CLASS	PRIOR	RTCODE	OCCURENCES	SUBTOTALS
A	04	CENTRE	215	
A	04	PHYSICS	126	
A	04	ARTS	75	
A	04	SCARBORDU	0	
A	04	ERINDALE	18	
A	04	DENTISTRY	0	
A	04	AEROSPACE	9	
A	04	BUSINESS	14	
A	04	WESTERN1	0	
A	04	WESTERN2	0	
A	04	CRJE	48	
A	04	TSO	6	
A	04	BATCH	0	
A	04	OTHERS	1	
				512
A	06	CENTRE	127	
A	06	PHYSICS	52	
A	06	ARTS	68	
A	06	SCARBORDU	5	
A	06	ERINDALE	2	
A	06	DENTISTRY	0	
A	06	AEROSPACE	3	
A	06	BUSINESS	10	
A	06	WESTERN1	2	
A	06	WESTERN2	0	
A	06	CRJE	13	
A	06	TSO	14	
A	06	BATCH	0	
A	06	OTHERS	13	
				309
A	08	CENTRE	41	
A	08	PHYSICS	1	
A	08	ARTS	8	
A	08	SCARBORDU	0	
A	08	ERINDALE	0	
A	08	DENTISTRY	0	
A	08	AEROSPACE	0	
A	08	BUSINESS	0	
A	08	WESTERN1	0	
A	08	WESTERN2	0	
A	08	CRJE	43	
A	08	TSO	5	
A	08	BATCH	0	
A	08	OTHERS	14	
				112
A	00	CENTRE	0	
A	00	PHYSICS	2	
A	00	ARTS	0	
A	00	SCARBORDU	0	
A	00	ERINDALE	0	
A	00	DENTISTRY	0	
A	00	AEROSPACE	0	
A	00	BUSINESS	0	
A	00	WESTERN1	0	
A	00	WESTERN2	0	
A	00	CRJE	0	
A	00	TSO	0	

A	00	BATCH	0
A	00	OTHERS	0
			2
A	OTHERS	CENTRE	0
A	OTHERS	PHYSICS	0
A	OTHERS	ARTS	0
A	OTHERS	SCARBOROU	0
A	OTHERS	ERINDALE	0
A	OTHERS	DENTISTRY	0
A	OTHERS	AEROSPACE	0
A	OTHERS	BUSINESS	0
A	OTHERS	WESTERN1	0
A	OTHERS	WESTERN2	0
A	OTHERS	CRJE	0
A	OTHERS	TSO	0
A	OTHERS	BATCH	0
A	OTHERS	OTHERS	0
			0
B	04	CENTRE	34
B	04	PHYSICS	9
B	04	ARTS	9
B	04	SCARBOROU	4
B	04	ERINDALE	0
B	04	DENTISTRY	0
B	04	AEROSPACE	0
B	04	BUSINESS	0
B	04	WESTERN1	0
B	04	WESTERN2	0
B	04	CRJE	19
B	04	TSO	0
B	04	BATCH	0
B	04	OTHERS	0
			75
B	06	CENTRE	27
B	06	PHYSICS	1
B	06	ARTS	1
B	06	SCARBOROU	7
B	06	ERINDALE	0
B	06	DENTISTRY	0
B	06	AEROSPACE	0
B	06	BUSINESS	0
B	06	WESTERN1	0
B	06	WESTERN2	0
B	06	CRJE	3
B	06	TSO	2
B	06	BATCH	0
B	06	OTHERS	1
			42
B	08	CENTRE	22
B	08	PHYSICS	2
B	08	ARTS	2
B	08	SCARBOROU	0
B	08	ERINDALE	0
B	08	DENTISTRY	0
B	08	AEROSPACE	0
B	08	BUSINESS	1
B	08	WESTERN1	0
B	08	WESTERN2	0
B	08	CRJE	22
B	08	TSO	23

B	08	BATCH	0
B	08	OTHERS	7
			79
B	00	CENTRE	0
B	00	PHYSICS	0
B	00	ARTS	0
B	00	SCARBOROU	0
B	00	ERINDALE	0
B	00	DENTISTRY	0
B	00	AEROSPACE	0
B	00	BUSINESS	0
B	00	WESTERN1	0
B	00	WESTERN2	0
B	00	CRJE	0
B	00	TSO	0
B	00	BATCH	0
B	00	OTHERS	0
			0
B	OTHERS	CENTRE	0
B	OTHERS	PHYSICS	0
B	OTHERS	ARTS	0
B	OTHERS	SCARBOROU	0
B	OTHERS	ERINDALE	0
B	OTHERS	DENTISTRY	0
B	OTHERS	AEROSPACE	0
B	OTHERS	BUSINESS	0
B	OTHERS	WESTERN1	0
B	OTHERS	WESTERN2	0
B	OTHERS	CRJE	0
B	OTHERS	TSO	0
B	OTHERS	BATCH	0
B	OTHERS	OTHERS	0
			0
C	04	CENTRE	10
C	04	PHYSICS	2
C	04	ARTS	2
C	04	SCARBOROU	1
C	04	ERINDALE	1
C	04	DENTISTRY	0
C	04	AEROSPACE	0
C	04	BUSINESS	5
C	04	WESTERN1	0
C	04	WESTERN2	0
C	04	CRJE	0
C	04	TSO	0
C	04	BATCH	0
C	04	OTHERS	7
			28
C	06	CENTRE	24
C	06	PHYSICS	1
C	06	ARTS	0
C	06	SCARBOROU	0
C	06	ERINDALE	0
C	06	DENTISTRY	0
C	06	AEROSPACE	0
C	06	BUSINESS	1
C	06	WESTERN1	0
C	06	WESTERN2	0
C	06	CRJE	11
C	06	TSO	0

C	06	BATCH	0
C	06	OTHERS	29
			66
C	08	CENTRE	3
C	08	PHYSICS	0
C	08	ARTS	0
C	08	SCARBORDU	0
C	08	ERINDALE	0
C	08	DENTISTRY	0
C	08	AEROSPACE	0
C	08	BUSINESS	0
C	08	WESTERN1	0
C	08	WESTERN2	0
C	08	CRJE	0
C	08	TSO	0
C	08	BATCH	0
C	08	OTHERS	9
			12
C	00	CENTRE	0
C	00	PHYSICS	0
C	00	ARTS	0
C	00	SCARBORDU	0
C	00	ERINDALE	0
C	00	DENTISTRY	0
C	00	AEROSPACE	0
C	00	BUSINESS	0
C	00	WESTERN1	0
C	00	WESTERN2	0
C	00	CRJE	0
C	00	TSO	0
C	00	BATCH	0
C	00	OTHERS	0
			0
C	OTHERS	CENTRE	0
C	OTHERS	PHYSICS	0
C	OTHERS	ARTS	0
C	OTHERS	SCARBORDU	0
C	OTHERS	ERINDALE	0
C	OTHERS	DENTISTRY	0
C	OTHERS	AEROSPACE	0
C	OTHERS	BUSINESS	0
C	OTHERS	WESTERN1	0
C	OTHERS	WESTERN2	0
C	OTHERS	CRJE	0
C	OTHERS	TSO	0
C	OTHERS	BATCH	0
C	OTHERS	OTHERS	0
			0
M	04	CENTRE	0
M	04	PHYSICS	0
M	04	ARTS	0
M	04	SCARBORDU	0
M	04	ERINDALE	0
M	04	DENTISTRY	0
M	04	AEROSPACE	0
M	04	BUSINESS	0
M	04	WESTERN1	0
M	04	WESTERN2	0
M	04	CRJE	0
M	04	TSO	0

M	04	BATCH	0	
M	04	OTHERS	0	
				0
M	06	CENTRE	0	
M	06	PHYSICS	0	
M	06	ARTS	0	
M	06	SCARBORDU	24	
M	06	ERINDALE	0	
M	06	DENTISTRY	0	
M	06	AEROSPACE	0	
M	06	BUSINESS	0	
M	06	WESTERN1	0	
M	06	WESTERN2	0	
M	06	CRJE	4	
M	06	TSO	0	
M	06	BATCH	0	
M	06	OTHERS	100	
				128
M	08	CENTRE	0	
M	08	PHYSICS	0	
M	08	ARTS	0	
M	08	SCARBORDU	0	
M	08	ERINDALE	0	
M	08	DENTISTRY	0	
M	08	AEROSPACE	0	
M	08	BUSINESS	0	
M	08	WESTERN1	0	
M	08	WESTERN2	0	
M	08	CRJE	0	
M	08	TSO	0	
M	08	BATCH	0	
M	08	OTHERS	24	
				24
M	00	CENTRE	0	
M	00	PHYSICS	0	
M	00	ARTS	0	
M	00	SCARBORDU	0	
M	00	ERINDALE	0	
M	00	DENTISTRY	0	
M	00	AEROSPACE	0	
M	00	BUSINESS	0	
M	00	WESTERN1	0	
M	00	WESTERN2	0	
M	00	CRJE	0	
M	00	TSO	0	
M	00	BATCH	0	
M	00	OTHERS	0	
				0
M	OTHERS	CENTRE	0	
M	OTHERS	PHYSICS	0	
M	OTHERS	ARTS	0	
M	OTHERS	SCARBORDU	0	
M	OTHERS	ERINDALE	0	
M	OTHERS	DENTISTRY	0	
M	OTHERS	AEROSPACE	0	
M	OTHERS	BUSINESS	0	
M	OTHERS	WESTERN1	0	
M	OTHERS	WESTERN2	0	
M	OTHERS	CRJE	0	
M	OTHERS	TSO	0	

M	OTHERS	BATCH	0
M	OTHERS	OTHERS	0
			0
W	04	CENTRE	0
W	04	PHYSICS	0
W	04	ARTS	0
W	04	SCARBORDU	0
W	04	ERINDALE	0
W	04	DENTISTRY	0
W	04	AEROSPACE	0
W	04	BUSINESS	0
W	04	WESTERN1	0
W	04	WESTERN2	0
W	04	CRJE	0
W	04	TSO	0
W	04	BATCH	0
W	04	OTHERS	0
			0
W	06	CENTRE	0
W	06	PHYSICS	0
W	06	ARTS	0
W	06	SCARBORDU	0
W	06	ERINDALE	0
W	06	DENTISTRY	0
W	06	AEROSPACE	0
W	06	BUSINESS	0
W	06	WESTERN1	0
W	06	WESTERN2	0
W	06	CRJE	0
W	06	TSO	0
W	06	BATCH	0
W	06	OTHERS	0
			0
W	08	CENTRE	0
W	08	PHYSICS	0
W	08	ARTS	0
W	08	SCARBORDU	0
W	08	ERINDALE	0
W	08	DENTISTRY	0
W	08	AEROSPACE	0
W	08	BUSINESS	0
W	08	WESTERN1	0
W	08	WESTERN2	0
W	08	CRJE	0
W	08	TSO	0
W	08	BATCH	0
W	08	OTHERS	0
			0
W	00	CENTRE	0
W	00	PHYSICS	301
W	00	ARTS	1287
W	00	SCARBORDU	51
W	00	ERINDALE	9
W	00	DENTISTRY	0
W	00	AEROSPACE	51
W	00	BUSINESS	0
W	00	WESTERN1	0
W	00	WESTERN2	0
W	00	CRJE	52
W	00	TSO	0

W	00	BATCH	981
W	00	OTHERS	59
			2791
W	OTHERS	CENTRE	0
W	OTHERS	PHYSICS	0
W	OTHERS	ARTS	0
W	OTHERS	SCARBOROU	0
W	OTHERS	ERINDALE	0
W	OTHERS	DENTISTRY	0
W	OTHERS	AEROSPACE	0
W	OTHERS	BUSINESS	0
W	OTHERS	WESTERN1	0
W	OTHERS	WESTERN2	0
W	OTHERS	CRJE	0
W	OTHERS	TSO	0
W	OTHERS	BATCH	0
W	OTHERS	OTHERS	0
			0
OTHERS	04	CENTRE	0
OTHERS	04	PHYSICS	1
OTHERS	04	ARTS	1
OTHERS	04	SCARBOROU	0
OTHERS	04	ERINDALE	0
OTHERS	04	DENTISTRY	0
OTHERS	04	AEROSPACE	0
OTHERS	04	BUSINESS	2
OTHERS	04	WESTERN1	0
OTHERS	04	WESTERN2	0
OTHERS	04	CRJE	0
OTHERS	04	TSO	0
OTHERS	04	BATCH	0
OTHERS	04	OTHERS	0
OTHERS	06	CENTRE	0
OTHERS	06	PHYSICS	0
OTHERS	06	ARTS	0
OTHERS	06	SCARBOROU	0
OTHERS	06	ERINDALE	0
OTHERS	06	DENTISTRY	0
OTHERS	06	AEROSPACE	0
OTHERS	06	BUSINESS	0
OTHERS	06	WESTERN1	0
OTHERS	06	WESTERN2	0
OTHERS	06	CRJE	0
OTHERS	06	TSO	0
OTHERS	06	BATCH	0
OTHERS	06	OTHERS	1
OTHERS	08	CENTRE	0
OTHERS	08	PHYSICS	0
OTHERS	08	ARTS	0
OTHERS	08	SCARBOROU	0
OTHERS	08	ERINDALE	0
OTHERS	08	DENTISTRY	0
OTHERS	08	AEROSPACE	0
OTHERS	08	BUSINESS	0
OTHERS	08	WESTERN1	0
OTHERS	08	WESTERN2	0
OTHERS	08	CRJE	0
OTHERS	08	TSO	0
OTHERS	08	BATCH	0
OTHERS	08	OTHERS	0

```
OTHERS     00         CENTRE              12
OTHERS     00         PHYSICS              5
OTHERS     00         ARTS                14
OTHERS     00         SCARBOROU            0
OTHERS     00         ERINDALE             0
OTHERS     00         DENTISTRY            0
OTHERS     00         AEROSPACE            0
OTHERS     00         BUSINESS             1
OTHERS     00         WESTERN1             0
OTHERS     00         WESTERN2             0
OTHERS     00         CRJE                 4
OTHERS     00         TSO                  0
OTHERS     00         BATCH                1
OTHERS     00         OTHERS               9
OTHERS     OTHERS     CENTRE               0
OTHERS     OTHERS     PHYSICS              0
OTHERS     OTHERS     ARTS                 0
OTHERS     OTHERS     SCARBOROU            0
OTHERS     OTHERS     ERINDALE             0
OTHERS     OTHERS     DENTISTRY            0
OTHERS     OTHERS     AEROSPACE            0
OTHERS     OTHERS     BUSINESS             0
OTHERS     OTHERS     WESTERN1             0
OTHERS     OTHERS     WESTERN2             0
OTHERS     OTHERS     CRJE                 0
OTHERS     OTHERS     TSO                  0
OTHERS     OTHERS     BATCH                0
OTHERS     OTHERS     OTHERS               0
THE TOTAL NUMBER OF RECORDS CLASSIFIED IS      4180


NAME          DESCRIPTION                    THIS DATA IS GENERATED BY LINE  34
FORMS         # FORMS MOUNTS
THE SUM OF FORMS  IS          0
        @EOJ                                                                  00000010

CORE USAGE          OBJECT CODE AND VARIABLE STORAGE=  1210 WORDS, TOTAL AVAILABLE=  1500 WORDS

DIAGNOSTICS         NUMBER OF ERRORS=      0, NUMBER OF WARNINGS=      4

                     FILE NUMBER         NUMBER OF RECORDS PROCESSED
                          5                    4231 (INPUT )
                     ANY FILES NOT LISTED DID NOT HAVE ANY RECORDS PROCESSED
```

BIBLIOGRAPHY

1. APL/360 Primer. IBM Form Number C20-1702.

2. APL PLUS File Subsystem Instruction Manual. Toronto: I.P. Sharp Associates Ltd., 1972.

3. Armor, D.J. "A Computer Language for the Analysis of Variance," reprinted from the 1969 Social Statistics Section of the American Statistical Association.

4. Armor, D.J. and Couch, A.S. Data Text Primer (final draft of the book published by The Free Press, New York 1972).

5. Bennett, R.K. "The Design of Computer Languages and Software Systems: A Basic Approach," Computers and Automation, (February 1969).

6. Class, A., et al. "Statistical Report Generator." A report of the University of Waterloo. April 1969.

7. Cress, P.H. and Dirksen, P.H. "SRG-40-Survey Report Generator." A report of the University of Waterloo, 1968.

8. Cress, P.H., et al. "Description of 1360 WATFOR. A Fortran IV Compiler." A Report of the Faculty of Mathematics, University of Waterloo, 1968.

9. Fortran IV (G) Program Logic Manual. IBM Form Number Y28-6638.

10. Fortran IV (H) Program Logic Manual. IBM Form Number Y28-0012.

11. Full American National Standard Cobol. IBM Form Number GC28-6396.

12. Genuys, F. (ed.) Programming Languages - Nato Advanced Study Institute. New York: Academic Press, 1968.

13. Rosen, S. "Programming Systems and Languages - A Historical Survey," in Programming Systems and Languages, ed. S. Rosen. New York: McGraw-Hill, 1967.

14. Sammet, J. Programming Languages: History and Fundamentals. Englewood Cliffs, New Jersey: Prentice Hall, 1969.

15. Shantz, P. W. et al. "WATFOR - The University of Waterloo Fortran IV COMPILER," Communications of the ACM, X, No. 1 (January 1967).