Betty Snell
Simon Fraser Computing Centre
Burnaby, British Columbia

PROGRAMMING LIBRARY APPLICATIONS IN PL/I

In February 1968, Simon Fraser University Computing Centre began extensive use of PL/I. To best explain what the library section of the Computing Centre has done with PL/I during the intervening period, I would like to begin by going back to 1968, explaining the problems with which we were faced, giving a brief description of the systems we had in operation at the time and of the systems we had planned.

In February 1968, we had five systems in operation: acquisitions system, serials listing system, catalog listing system, circulation system, and books inventory system (Figure 1). All systems had been programmed in 1401 Autocoder, running on an IBM 360/40 tape and disk system under emulator.

Running under emulator was only a temporary measure. All programs would have to be rewritten to make effective use of the operating system and since a 360/50 without emulator was due to arrive in December 1968, all programs would have to be converted in ten months.

However, in most cases this was not to be a one-to-one conversion of programs. In most systems so many modifications had been made and programs added that the systems needed redesigning. In some cases programs could be combined; in others the systems needed to be expanded to allow greater flexibility and additional lists.

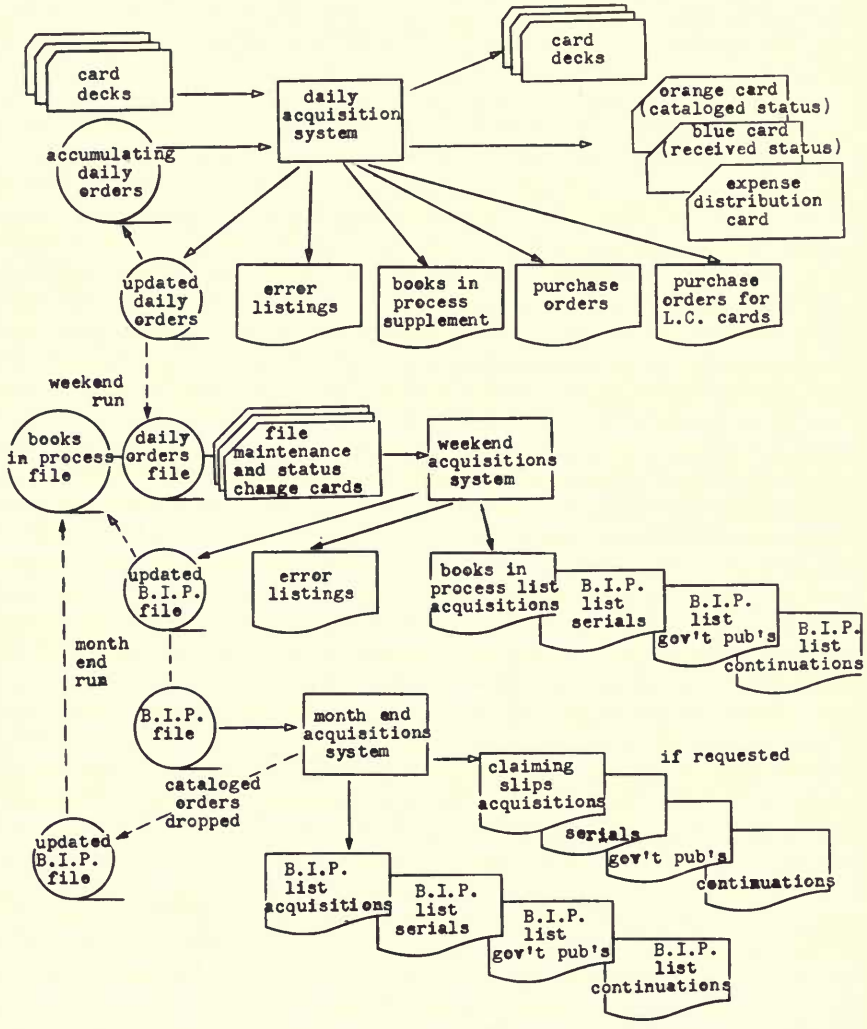
In addition to the conversion and overhaul of existing systems, three new major systems had been planned. These were the out-of-print desiderata system, maps listing system, and pamphlets listing system.

SIMON FRASER UNIVERSITY LIBRARY APPLICATIONS

Acquisitions System

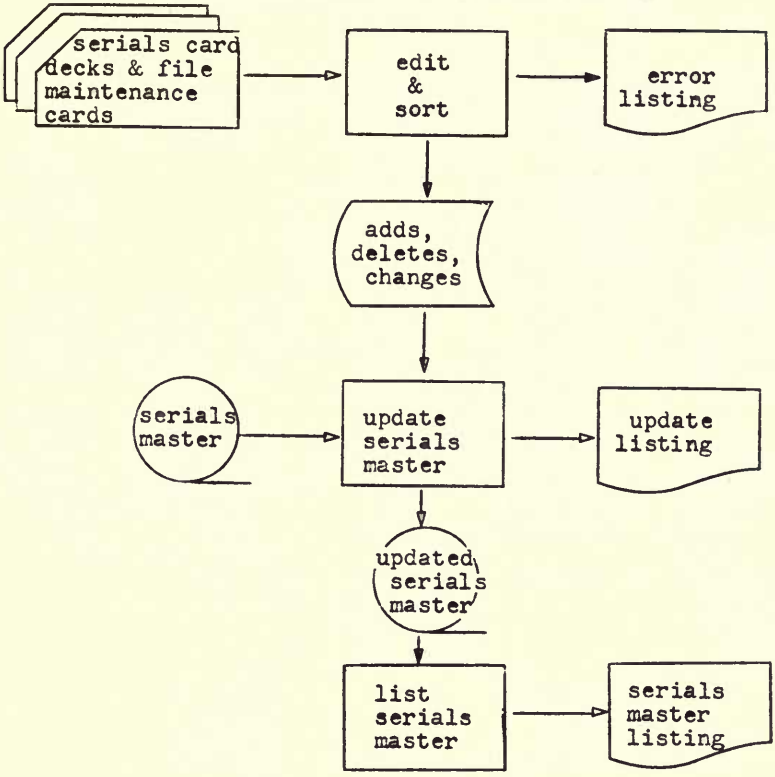
The automated acquisitions system went into production in April, 1966, seven months after the university opened. Although it has been modified and expanded many times since its inception, the basic design remains the same.

Figure 1 – System Flowchart: Acquisitions



Once the requests have been searched, the following information is punched on cards: purchase order number, purchase order date, account number, estimated time of arrival, estimated price, Library of Congress card number, number of copies, vendor number, complete main entry in Library of Congress format, short title, complete imprint, edited entry, edited title and purchase order comments. These cards or order decks are sent to the computing centre and serve as input in the daily acquisitions system (now run only three times a week).

Figure 1 – System Flowchart: Serials Listing



SPECIAL LISTS

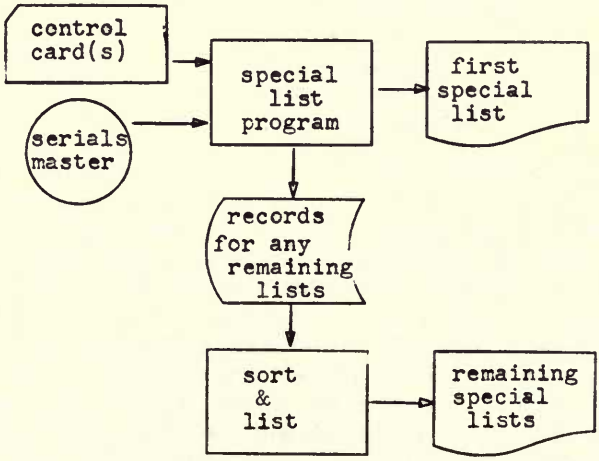
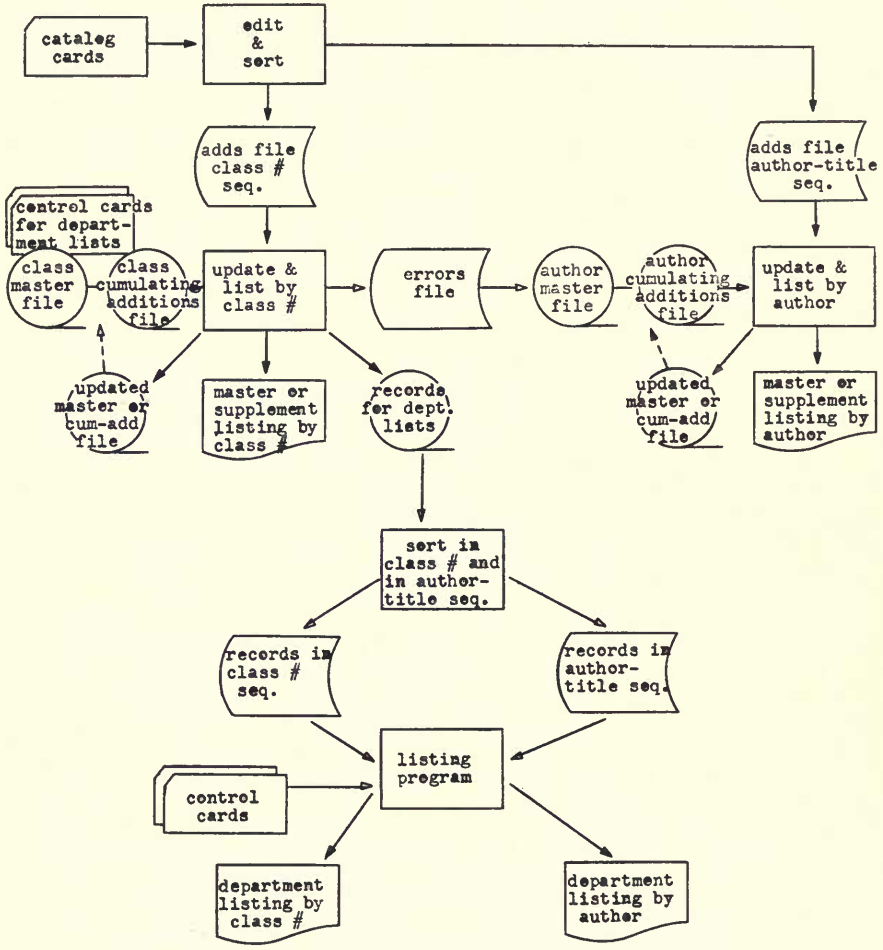
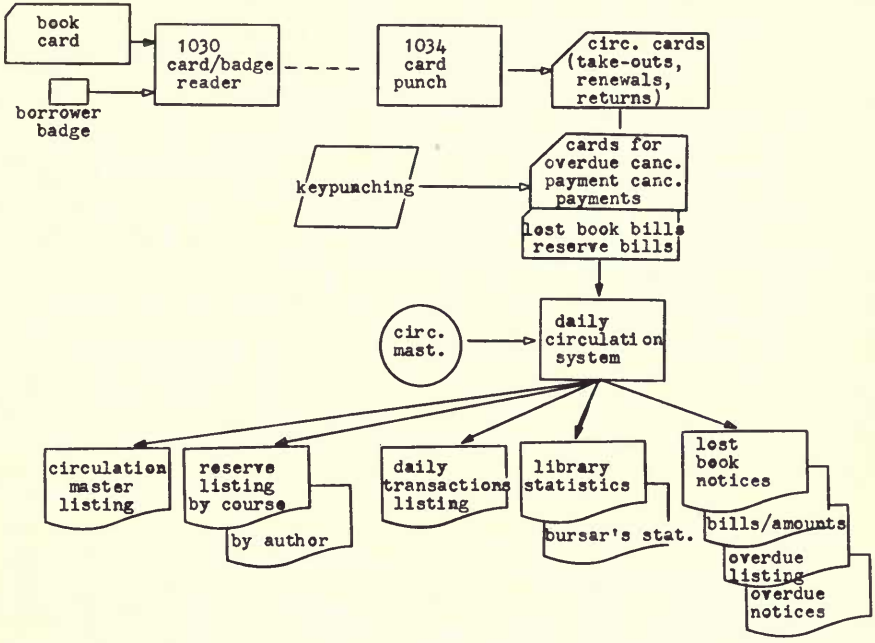


Figure 1 – System Flowchart: Catalog Listing



The following fields on each deck are checked for validity: vendor number, currency code, number of copies, type of acquisitions, purchase order number, amount, account number, and title (duplication check). For each valid order the following are produced: a purchase order, a purchase order for Library of Congress cards, a blue card to be returned when the book is received, an orange card to be returned when the book is cataloged and an expense distribution card. When the book is received the price is punched into the expense distribution card which is then sent to the accounting department. An error listing for invalid orders and a supplement to the books-in-process list are printed with each daily acquisitions run. This supplement listing contains all valid orders processed for the week.

Figure 1 – System Flowchart: Circulation



Each week the books-in-process file is updated with the new orders for the week, file maintenance cards, and status change cards (blue and orange). A new books-in-process list is produced with separate lists for acquisitions, continuations, serials, and government publications.

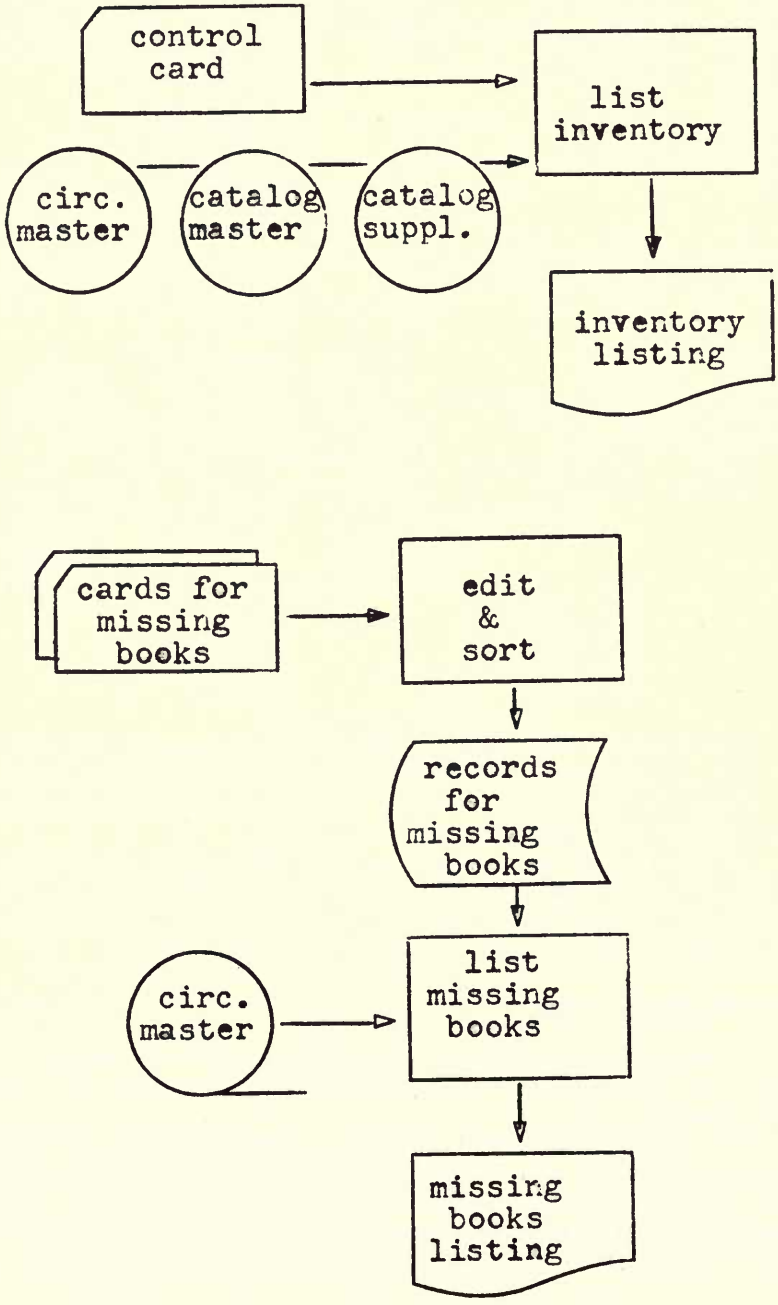
At the end of the month, cataloged orders are dropped from the book-in-process file, and lists of overdue orders and claiming slips, if requested, are produced. At present the acquisitions system is being studied with the aim of making it a more compact and efficient system.

Cataloging Listing System

Once an order has been cataloged, the book circulation card is punched, a duplicate card is sent to the Computing Centre to form part of the library catalog files.

The catalog listing system provides a master catalog in classification number sequence and in author and title sequence every four months. Supplement listings cumulating for four months are produced weekly. Lists for departments in each sequence are printed each month as well as when the master is printed.

Figure 1 — System Flowchart: Books Inventory



Circulation System

When a borrower wants to check out or renew a book, he presents his borrower badge (identification card) and the book to the IBM 1031 operator. The book card and borrower badge are fed into the IBM 1031 badge reader. The information in the badge and book card is transferred to the remote 1034 card punch, and a circulation card is punched.

When a book is returned, the book card and return badge are fed through the 1031 reader and a corresponding card is punched by the 1034. Cards from books placed on reserve are fed through the 1030 system along with a course badge. Thus reserve books appear on the circulation master listing as well as on separate reserve listing by course and by author.

The above cards together with fine payment cards and fine cancellation cards are used to update the circulation master file daily. The following reports are produced from the circulation files: daily—daily transactions listing, circulation master listing, bills and overdue notices, circulation statistics, and bursar's statistics; weekly—reserves by course, reserves by author, Xerox reserves; upon request—bills and amounts listings. At present the circulation system is being redesigned and programmed for an on-line system.

On-Line Loans System

It is proposed to implement the on-line loans system in two stages or phases. Basically the system is designed to facilitate inquiries to the master loan file and to update this file.

Phase 1. Equipment for Phase 1 will include three 2260 display terminals, three 1031 badge/card readers, and a 1034 card punch. The file will be held on one 2311 disk drive. The on-line system will be operating from 8:00 A.M. until 2:00 A.M. Two inquiry terminals will be available for library users and one for staff use.

The principal feature of Phase 1 will be direct inquiry to the master loan file regarding the current status of books. This will render the master loan listing of the current system unnecessary, with the exception of reserve books, which will still need to be listed at this stage.

Output from the 1034 card punch will be used to update the master file during the day in batches, probably every one or two hours. After brief editing, the cards will be placed into update areas on the master file. These update areas are also to be scanned by the inquiry program.

Staff members will, in addition to making book status queries, be able to place a hold on a book using the 2260 display terminal. Also they will be able to enter renewals and make inquiries for billing information. A 100 position record is used containing classification number, author/title, due date, return date if a bill, the borrower code and number, the amount if a fine, and special flags for the type of record. The master loan file uses tables in core and on disk similiar to indexed sequential organization (which was not used because of the large numbers of legitimate duplicate records). Each track will contain twenty-eight master file records and eight areas available for updates during the day. The file is to be reorganized nightly.

In accessing file records during the 2260 inquiry program, the copy number, volume number and date of publication are ignored. This is to make available to the user the status of as many copies of the required title as possible.

Communication with 2260 display units is facilitated through the use of an IBM written set of assembler modules, PGAM; a PL/I interface to OS/360 GAM. This uses a series of CALL statements to reference various entry points in one module of the PGAM system, which is linkage-edited with the PL/I program.

The PL/I program must either consist of a single task, or it must create a sub-task for each display station with which it communicates. If the program consists of a single task, dispatching of program control between the various display stations is the responsibility of the PL/I program. At the time of writing, it has not been decided whether multi-tasking or single-tasking will best meet the needs of speed and core economy.

Phase II. In Phase II there will be an additional 2260 display unit, two 1033 printers, and two 1031 badge/card readers (one for reserves, one for Xerox reserves). In addition to the operations performed in Phase I, Phase II will include direct updating of the master loan file from the 1031 readers.

An abbreviated catalog file will be held on disk in order to be able to indicate whether other copies of the required book are in the stacks. Automatic fining of reserves will be introduced. A facility will be built in to prevent use of the library by those not possessing valid borrower numbers or to exclude particular borrower numbers.

The master loan file, now containing reserve books and Xerox reserves, will be held on a 2314 disk drive. The target date for Phase II is November 1969.

Library Inventory System

The library inventory system is a by-product of the catalog and circulation systems designed to produce a listing of missing books. Books on the catalog file which are not on the circulation file should be on the shelves.

The librarian decides on the range of books to be checked, and a control card with the first two positions of the beginning classification number and the last classification number and the last classification number is punched, e.g., B-BC. The circulation master file is passed against the catalog master and supplement file and a shelflist is produced.

Missing books are checked off the list and cards for these books are punched. The next day these cards are passed against the new circulation master and a listing of those books still missing is produced. This list is checked against the shelves again and cards for those still missing are sent to the computing centre. Another check is made against the circulation master and a final list of missing books produced. This listing is used for reordering as well as for statistics on lost books.

Serial Listing System

Card decks containing title, holdings, notes, serial number, department code, and location were originally sent to the Computing Centre weekly. Each week these decks were edited, sorted, passed against the serials master, and an update listing and a master listing produced. Lists by department were available upon request. However, a more flexible file maintenance procedure was necessary since holdings were constantly changing. And, since all programs had to be rewritten for conversion anyway, more information was added to each record and additional listing programs were written.

In addition to the weekly update and master listings, month-end lists of new titles ordered, new titles received and backfiles received are now produced. Special listings are printed on a request basis. The content of each list is determined by the following variables: subjects (up to thirty-five), publisher type, class, form, special collections, location, status of order, status of journal, and department code (up to nine). One card is coded for each separate list.

Out-Of-Print Desiderata System

Once a title has been searched and the book found to be out of print, the book request card and bibliographic information is sent to the out-of-print unit of acquisitions. A slip containing a dummy purchase order number, estimated price, country code, account number, number of copies and volumes, up to ten subject codes, and any desired comment is attached to the request card. From all of this information, a deck of cards is punched. These decks are sent to the Computing Centre weekly.

The decks are sorted, edited, and a tape record produced. This tape is used to update the master file. Upon request, desiderata slips are produced on 8½" x 3½" four-part forms. A card containing country code and up to ten subject codes determines each list. Up to ten lists can be produced weekly.

When a quotation for a book is received, the vendor number and actual price are punched into the deletion card (first card of deck). If for any other reason a title should be deleted, a *D* is punched in the vendor number field of the deletion card. Deletion cards are sent to the Computing Centre weekly.

The following reports are produced weekly: master listing and statistical report by account, or supplement listing, deletion statistical reports by vendor and account, and desiderata slips.

Pamphlet Listing System

The information necessary to list a pamphlet is gathered and punched on a deck of cards. Each card in the deck carries an identification or deck number, a card type and a card code. This allows easy sorting into sequence and checking for duplicates. Card type one is the general information card and carries the file maintenance code (A-addition, D-deletion, C-change), quality code (indicating whether or not the pamphlet is ephemeral), publication date, call number and location. Card type two is the subject card. There may be a

maximum of ninety-nine subject cards per deck, meaning that the information for a pamphlet may appear under as many as ninety-nine different subject headings. Card type three is the title card, four the entry card, and five the imprint card. There may be up to two of each of these. Three times a year these decks are sent to the Computing Centre, sorted, edited and one variable length disk record per subject code produced. This file is sorted by title within subject and used to update the master file. The master file is listed, and an index to the list produced.

Maps Listing System

The maps listing system is similar to the pamphlet listing system. Both use "strictly variable length" records and both use exploded files. The maps system creates one variable length area record from the information contained on card decks and as many subject records as there are subject codes per deck.

The area file and the subject file are updated and listed every four months. The area master file is sorted by alphabetic-area sequence also.

PL/I

All in all, over ninety major programs were originally written in PL/I, tested, documented and put into production in one year. This figure does not count "one-shot" jobs such as programs to convert 1401 files. Moreover, these programs were written by three programmers who had no previous PL/I experience. Only one had a PL/I course, and this was at the time of the first version of PL/I.

Simon Fraser University was the first major installation in the Vancouver area to use PL/I. Thus it was often difficult to get direct answers to questions regarding programming problems. But with time and experience these problems have disappeared. IBM worked out many of the bugs and fixed them in later releases. However, the new releases have posed a problem in some cases. Program statements which would compile on earlier releases now produce warning and error messages when recompiled under the newer releases. In spite of all of the problems, the job was done. On the average a program was written, compiled, tested, documented and put into production in less than two weeks. And in many cases system design and program revision are included in this time. I think that this short implementation time can best be explained by taking a look at PL/I itself.

PL/I is a high-level programming language designed for both commercial and scientific applications and for real-time and systems applications. Although it looks like a combination of FORTRAN, (a scientific programming language), and COBOL, (a commercial programming language), it is much more flexible than either.

I do not want to go into a detailed comparison of PL/I and COBOL. For the average job, either language will do. However, I think PL/I is more flexible and faster to code and to de-bug. Moreover, release 16 and version 4 support multi-tasking which allows simultaneous service to remote terminals for on-line systems.

Many features of PL/I contribute toward making it an easy language to learn and to use. First, it is not necessary to know all aspects of PL/I to write a program. You may use sub-sets of PL/I and ignore other features of the language. As always there are many ways to write a routine to do a specific task, some more costly in core space and running time than others, and in PL/I there is usually at least one method which is easy on the programmer. Built-in functions, default attributes, free format, and sophisticated control statements save the programmer much tedious coding. Liberal use of comments makes programs easy to follow.

Most programming jobs for the library consist of routines for the following: 1) editing cards/card decks and creating tape/disk records; 2) updating and listing files; and 3) producing statistical reports.

I will give a few examples of these routines in PL/I. The first two are not meant to be complete programs, nor are they meant to be the best methods for handling the job; they do however, work.

If we consider a serials card deck as shown in Figure 2 we will first need routines to check that the serial number or ID number, card type, and card code are numeric, and that the subject field and title are not blank. Input will be in sequence: serial number-card type-card code. If the above information is all right we will create a variable length tape record, as shown in Figure 3. Otherwise we will print the appropriate error message and bypass the rest of the deck.

To begin, it is not necessary to define or declare the card input as long as one uses the default file name SYSIN. Next we need an input area for the card file. We can set up a structure as follows in Figure 4.

Let us consider another example: suppose on our out-of-print master file we carry account number, estimated price, and a field indicating whether or not a desiderata slip has been sent out for each title. We want to produce a report. There are fifty-three different accounts, so the first thing we do is set up arrays for account numbers and for account names, each containing fifty-three elements. Then we set up a two-dimensional array 53 x 4 to contain the variable information. A separate array four elements long may be set up for totals. Since the account numbers range from 901-999, a cross reference table ninety-nine elements long can be set up to give the corresponding row in the two-dimensional array. For example, 904 is the second account to be listed, so the fourth element in the ninety-nine element array will be 2. Zeros can be used for cases where there is no account number. For example, there is no account number 902; so the second element in the ninety-nine element array will be zero. Figure 5 is an example of how this might be programed.

Before going on to my last example, I would like to discuss our use of variable length records. First it was decided to use variable length records wherever it was necessary to carry complete information but where the amount of this information could vary greatly. Variable length records make more efficient use of storage space (tape/disk) and hence allow faster time to read the entire file.

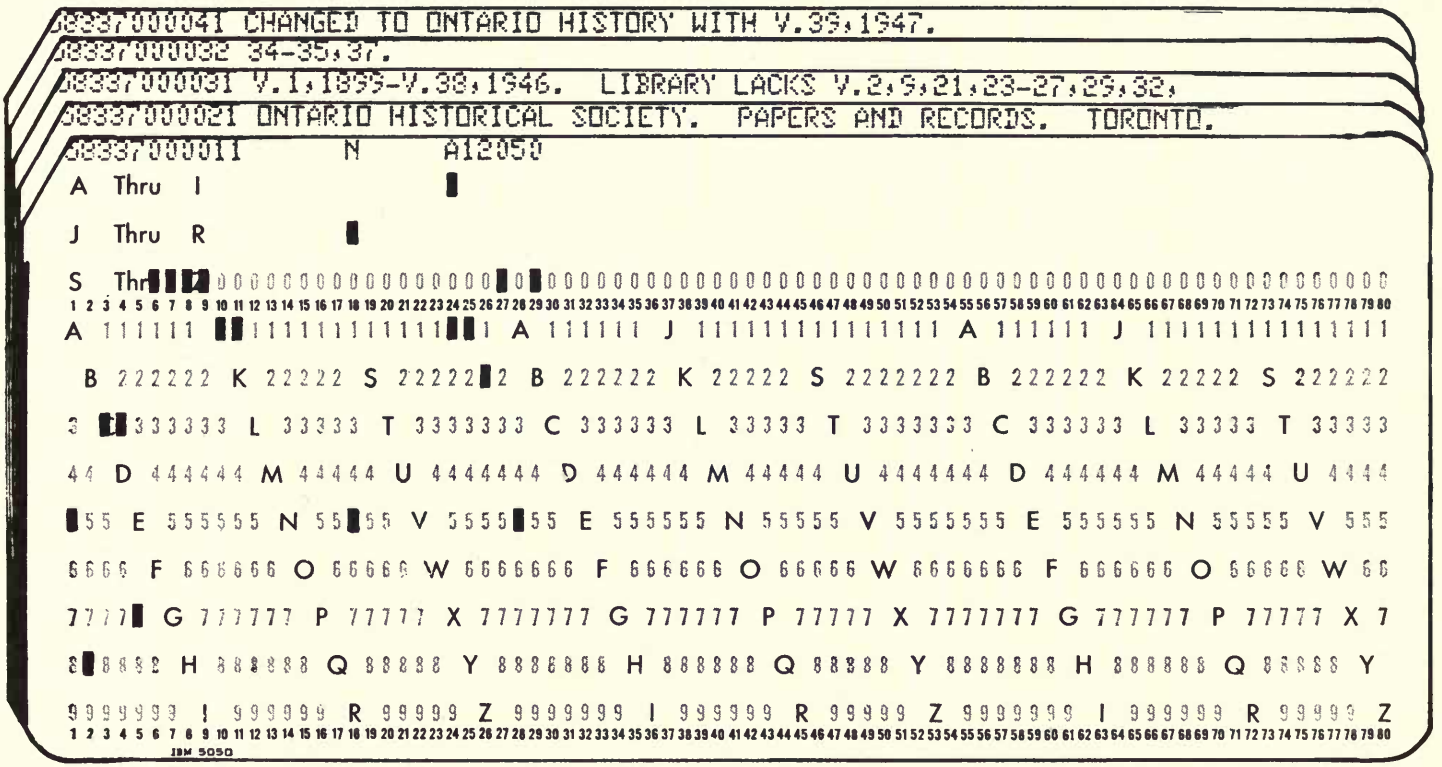


Figure 2 – Serials Card Deck

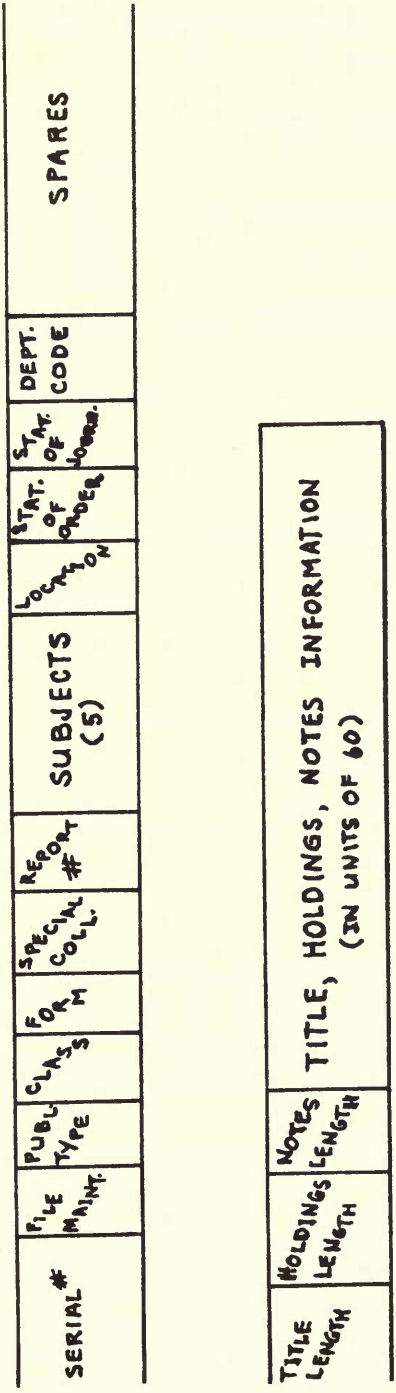


Figure 3 — Variable Length Tape

Figure 4 — Structure Set-up and Program of an Input Area for the Card File

```
                                                                    */
      IF CARD_TYPE='1' & INAREA.SUBJECT= ' ' THEN DO;
                  PUT EDIT('SUBJECT CODE MISSING FOR ',
                        INAREA.SERIAL#)(SKIP,A);
                  GO TO REJECT;
                                                  END;
/**CHECK FOR TITLE
                                                                    */
      IF CARD_TYPE='2' AND DATA=(80)' ' THEN DO;
                  PUT EDIT('TITLE MISSING FOR ',INAREA.SERIAL#)
                        (SKIP,A);
                  GO TO REJECT;
                                                  END;
/**THAT TAKES CARE OF THE EDITING. NOW WE WANT TO CREATE THE
   VARIABLE LENGTH TAPE RECORD.
                                                                    */
      DECLARE TAPEOUT FILE OUTPUT RECORD ENV(VB 2416 4096);
      DECLARE 1 OUTAREA,
                2 SERIAL#       CHAR(9),
                2 FILE_MAINT    CHAR(1),
                2 PURL_TYPE     CHAR(1),
                2 CLASS         CHAR(1),
                2 FORM          CHAR(1),
                2 SPEC_COLL     CHAR(1),
                2 REPORT#       CHAR(1),
                2 SUBJECT       CHAR(1),
                2 LOCATION      CHAR(1),
                2 STAT_ORDER    CHAR(1),
                2 STAT_JOURNAL  CHAR(1),
                2 DEPT          CHAR(3),
                2 SPARES        CHAR(45),
                2 TLENGTH       PIC'999', /*LENGTH OF TITLE FIELD    */
                2 HLENGTH       PIC'999', /*LENGTH OF HOLDINGS FIELD */
                2 NLENGTH       PIC'999', /*LENGTH OF NOTES FIELD    */
                2 VARIABLE      CHAR(1720);/*TITLE,HOLDINGS,NOTES    */
      DCL OUTAREA_RD CHAR(80) DEF OUTAREA;
/** MOVE INFORMATION FROM FIRST CARD
                                                                    */ 4
      IF CARD_TYPE='1' THEN OUTAREA=INAREA, BY NAME;4
                  ELSE DO;                                              This will move all fields with some
                  SUBSTR(VARIABLE,I,60)=DATA;                             name from INAREA to OUTAREA
                  I=I+60;
                  KOUNTER(CARD_TYPE)=KOUNTER(CARD_TYPE)+1;
            /** THIS COUNTS THE NUMBER OF CARDS FOR EACH CARD TYPE  */
                        END;
      GO TO READ;
REJECT: CHECK='REJECT';
      GO TO READ;
END_OF_DECK: IF CHECK='REJECT' THEN GO TO INITIALIZE;
            TLENGTH=KOUNTER(2)*60;
            HLENGTH=KOUNTER(3)*60;
            NLENGTH=KOUNTER(4)*60;
            LENGTH_REC=80+TLENGTH+HLENGTH+NLENGTH;
            CALL WRITE;  /* PROCEDURE TO WRITE RECORD */
INITIALIZE:
            GO TO START; /* BEGIN PROCESSING NEXT DECK */
WRITE: PROCEDURE;
      DCL OUTAREA2 CHAR(LENGTH_REC);
      OUTAREA2=OUTAREA_RD;

      WRITE FILE (TAPEOUT) FROM (OUTAREA2);
      END;
RF1:  FORMAT(SKIP(2),COL(10),A,A);
```

The basic format of variable length records is as follows: a fixed portion containing general information, a counter for each variable length field, and variable length fields; however, the variable length fields themselves may differ in format. The next sample program will show how to create two different kinds of variable length records from a serials card deck and two different ways to list these records.

The first method adds fixed-length (sixty characters) trailers to the fixed portion of the record, one trailer for each title, holdings, or notes card. This is the method we use in our serials system. The second method adds exactly as much information to the fixed portion of the record as appeared on the cards. For example, if the title was only forty-three characters long, then only forty-three characters would be stored on tape rather than sixty.

The first method allows us to list sixty characters a line very simply. With the second method, more complicated print routines must be used in order to avoid splitting words at the end of each print line.

Figure 4 (continued)

```
      DECLARE 1 INAREA,
                2 SERIAL#       CHAR(9),
                2 CARD_TYPE     CHAR(1),
                2 CARD_CODE     CHAR(1),
                2 FILE_MAINT    CHAR(1),
                2 PUBL_TYPE     CHAR(1),
                2 CLASS         CHAR(1),
                2 FORM          CHAR(1),
                2 SPEC_COLL     CHAR(1),
                2 REPORT#       CHAR(1),
                2 SUBJS(5)      CHAR(1),
                2 SPARE         CHAR(1),
                2 LOCATION      CHAR(1),
                2 STAT_ORDER    CHAR(1),
                2 STAT_JOURNAL  CHAR(1),
                2 DEPT          CHAR(3),
                2 EXTRAS        CHAR(51);
/*
   THIS ALLOWS US TO ACCESS EACH FIELD ON CARD TYPE 1 SEPARATELY.  FOR
   ALL REMAINING CARDS WE NEED ONLY DECLARE AN AREA 60 CHAPACTERS LONG
   BEGINNING IN COL. 13. WE MAY DO THIS BY MEANS OF AN OVERLAY AS
   FOLLOWS:
                                                                    */
      DCL DATA CHAR(60) DEFINED INAREA POS(13);
/**THUS ALL CARDS CAN BE READ INTO THE SAME AREA - INAREA
                                                                    */
/**SET UP AND INIATIALIZE ALL FIELDS TO BE USED LATER
                                                                    */
  READ: READ FILE (SYSIN) INTO (INAREA);
/**IS THIS THE FIRST CARD?
                                                                    */
       IF FIRST=1 THEN DO;1                                              1
            FIRST=0;
            PREVIOUS#=INAREA.SERIAL#;/* NEEDED TO CHECK FOR END OF     */
            GO TO START;            /*                          DECK  */
                 END;
/**ARE WE AT THE BEGINNING OF A NEW DECK?
                                                                    */
       IF INAREA.SERIAL# > PREVIOUS# THEN GO TO END_OF_DECK;
/**ARE WE REJECTING THIS DECK?
                                                                    */
       IF CHECK='REJECT' THEN GO TO READ;
/**IS SERIAL NUMBER NUMERIC?
                                                                    */
  START:    DO I=1 TO 9;
       IF SUBSTR(INAREA.SERIAL#,I,1)2 < '0'3THEN DO;                     2
                 PUT EDIT('INVALID SERIAL# - ',INAREA.SERIAL#)
                      (R(RFI));  /* REMOTE FORMAT STATEMENT        */
                 GO TO REJECT;                                          3
                                        END;
            END;
/**ARE CARD TYPE AND CARD CODE NUMERIC?
                                                                    */
       IF CARD_TYPE < '0' | CARD_CODE < '0' THEN DO;
                 PUT EDIT('INVALID CARD TYPE OR CARD CODE FOR ',
                          INAREA.SERIAL#)(R(RFI));
                 GO TO REJECT;
                                          END;
/**IS THERE AT LEAST ONE SUBJECT CODE?
```

1 If the condition for the IF-THEN DØ does not hold then go to the corresponding END statement

2 SUBSTR(X,I,J) is a reference to field X starting in position I, J characters long

3 The collating sequence is special characters < blanks < alphabetics < numbers. Thus anything 0 is not numeric

Figure 5 – Program Containing Account Number, Account Names, Variable Information, and Totals

```
DECLARE ACCTNOS(53) PIC'999' INIT
   ((1)'901',(1)'904',(1)'905',(1)'906',(1)'907',(1)'908',
    (1)'909',(1)'910',(1)'912',(1)'913',(1)'914',(1)'915',
    (1)'916',(1)'917',(1)'918',(1)'919',(1)'920',(1)'921',
    (1)'922',(1)'923',(1)'924',(1)'928',(1)'930',(1)'935',
    (1)'940',(1)'942',(1)'944',(1)'946',(1)'950',(1)'951',
    (1)'952',(1)'960',(1)'965',(1)'970',(1)'975',(1)'978',
    (1)'979',(1)'980',(1)'982',(1)'984',(1)'986',(1)'987',
    (1)'988',(1)'989',(1)'990',(1)'991',(1)'992',(1)'993',
    (1)'994',(1)'995',(1)'996',(1)'997',(1)'999');
DECLARE ACCTNAMES(53) CHAR(23) INIT
   ((1)'ECONOMICS AND COMMERCE ',
    (1)'MEDIEVAL               ',
    (1)'ENGLISH                ',
    (1)'SHAKESPEARE            ',
    (1)'CONTEMP LIT COLLECTION ',
    (1)'17TH CENTURY           ',
    (1)'18TH CENTURY           ',
    (1)'GEOGRAPHY              ',
    (1)'COMMONWEALTH           ',
    (1)'19TH CENTURY BRIT.     ',
    (1)'19TH CENTURY U.S.A.    ',
    (1)'20TH CENTURY BRIT.     ',
    (1)'20TH CENTURY U.S.A.    ',
    (1)'ENGLISH LANGUAGE       ',
    (1)'MOD LANG-FRENCH        ',
    (1)'         -GERMAN       ',
    (1)'         -RUSSIAN      ',
    (1)'         -SPANISH      ',
    (1)'         -LINGUISTICS  ',
    (1)'         -OTHER        ',
    (1)'         -LIT          ',
    (1)'PHILOSOPHY             ',
    (1)'P.S.A.                 ',
    (1)'PSYCHOLOGY             ',
    (1)'PROF FOUND CENTRE      ',
    (1)'SOCIAL & PHIL FOUND    ',
    (1)'BEHAVIORAL SCIENCE FOUN',
    (1)'PHYSICAL DEVELOPMENT   ',
    (1)'COMMUNICATIONS-FINE ART',
    (1)'               -PERF ART',
    (1)'               -COMMUNIC',
    (1)'BIOLOGICAL SCIENCES    ',
    (1)'CHEMISTRY              ',
    (1)'MATHEMATICS            ',
    (1)'PHYSICS                ',
    (1)'LIBRARY-COLLECTION     ',
    (1)'         -LIBRARY      ',
    (1)'         -NAT BIBLIOGRAPH',
    (1)'         -HUMANITIES   ',
    (1)'         -SOCIAL SCIENCES',
    (1)'         -SCIENCE      ',
    (1)'         -MAPS         ',
    (1)'HISTORY-LAT AMERICA    ',
    (1)'         -CANADA       ',
    (1)'         -U.S.A.       ',
    (1)'         -AFRICA       ',
    (1)'         -NEAR EAST    ',
    (1)'         -FRANCE       ',
```

Figure 5 (continued)

```
          (1)'       -GERMANY        ',
          (1)'       -U.S.S.R.       ',
          (1)'       -GRT BRITAIN    ',
          (1)'       -OTHER          ',
          (1)'OTHERS                 ');
/** SET UP CROSS REFERENCE TABLE                                        */
      DCL TABLE(99) PIC'99' INIT
         ((1)'01',(2)(1)'00',(1)'02',(1)'03',(1)'04',(1)'05',(1)'06',
          (1)'07',(1)'08',(1)'00',(1)'09',(1)'10',(1)'11',(1)'12',
          (1)'13',(1)'14',(1)'15',(1)'16',(1)'17',(2)'18',(1)'19',
          (1)'20',(1)'21',(3)(1)'00',(1)'22',(1)'00',(1)'23',
          (4)(1)'00',(1)'24',(4)(1)'00',(1)'25',(1)'00',(1)'26',
          (1)'00',(1)'27',(1)'00',(1)'28',(3)(1)'00',(1)'29',(1)'30',
          (1)'31',(7)(1)'00',(1)'32',(4)(1)'00',(1)'33',(4)(1)'00',
          (1)'34',(4)(1)'00',(1)'35',(2)(1)'00',(1)'36',(1)'37',
          (1)'38',(1)'00',(1)'39',(1)'00',(1)'40',(1)'00',(1)'41',
          (1)'42',(1)'43',(1)'44',(1)'45',(1)'46',(1)'47',(1)'48',
          (1)'49',(1)'50',(1)'51',(1)'52',(1)'00',(1)'53');
/** SET UP TWO-DIMENSIONAL ARRAY FOR VARIABLES                          */
      DCL ARRAY(53,4) DEC(6,2) INIT((255)0);
      DCL TOTALS(4)   DEC(8,2) INIT((4)0);
/** DECLARE INPUT FILE AND INPUT AREA
 READ: READ FILE (MASTER) INTO (INAREA);
      J=SUBSTR(ACCT,2,2);  /**PICK UP LAST TWO POSITIONS OF ACCT#    */
      I=TABLE(J);
      IF I=0 THEN I=53;    /**IF NO SUCH ACCT# PUT IT IN 'OTHERS'    */
      ARRAY(I,1)=ARRAY(I,1)+1; /**ADD 1 TO NO. TITLES ON MASTER      */
      ARRAY(I,2)=ARRAY(I,2)+ESTPRICE; /**ADD ESTIMATED AMOUNT        */
      IF ON_SEARCH=1 THEN DO;          /*IF THE TITLE IS ON SEARCH   */
          ARRAY(I,3)=ARRAY(I,3)+1;
          ARRAY(I,4)=ARRAY(I,4)+ ESTPRICE;
                          END;
      GO TO READ;
/**** WHEN END OF FILE IS REACHED ON THE MASTER TAPE
      GO INTO THE FOLLOWING PRINT ROUTINE                               */
            DO I=1 TO 53;
      PUT EDIT(ACCTNOS(I),ACCTNAMES(I),(ARRAY(I,*))
              (COL(28),P'999',X(5),A,X(1),P'ZZZ,ZZ9',X(5),
               P'ZZZ,ZZZV.99',X(6),P'ZZZ,ZZ9',X(5),P'ZZZ,ZZZV.99');
            END;
/**** CALCULATE AND PRINT TOTALS                                       */
            DO I=1 TO 4;
            DO J=1 TO 53;
      TOTALS(I)=TOTAL(I)+ARRAY(J,I);
            END;
            END;
      PUT EDIT('TOTALS',(TOTALS(*)))(SKIP(2),COL(43),A,COL(61),
               P'ZZZ,ZZ9',X(2),P'ZZ,ZZZ,ZZZV.99',X(6),P'ZZZ,ZZ9',
               X(2),P'ZZ,ZZZ,ZZZV.99');
                                                                    */
```