B A S I C F O R I N S T R U C T I O N A L U S E

by

James P. Schwar
Computer Center
Lafayette College
Easton, Pennsylvania

February, 1975

ABSTRACT

3000 BASIC serves as the introductory programming language for the arts, sciences and engineering at Lafayette College. The self-teaching (interpretive) nature of the language, coupled with its power and versatility makes BASIC a natural choice for the first programming course. The tutor programs, upgraded and expanded from 2000 BASIC, are an integral part of the course. Two, three-credit courses are offered in introductory programming, one for engineers, the other for science and liberal arts students. The HP-3000, through terminal access, provides "hands-on" experience for each student. A minimum of four programming problems are required from each participant in the courses, with computer-output mandated for each assignment. Course syllabi are included.

INTRODUCTION

An introductory course in computer programming should

1. acquaint the student with the fundamentals of computer programming,
2. instill an appreciation of computing and computing applications,
3. direct the student toward a logical solution of problems via flow-charting and programming,
4. reinforce the above concepts through the writing of computer programs.

BASIC (Beginners All-Purpose Symbolic Instruction Code) was selected as the primary programming language due to its interpretive nature. The ability to interact through the language is invaluable in an instructional environment. This benefit more than offsets the slow execution of interpreted programs. Experience has also shown that the BASIC interpreter does not generate the machine loading that results from a compiler (such as Fortran). This would not necessarily be the case in a production environment. The 3000 BASIC language is in itself a very powerful superset of Dartmouth BASIC and is equal to if not superior to Fortran for programming capability.

COURSE ORGANIZATION

Two distinct student types must be instructed in programming. The engineering and the science/liberal arts student. Programming for the engineer is incorporated within a 3 credit, second semester engineering science course. This course is divided into three, 1-credit parts: two-dimensional statics, programming (figure 1), and vector statics. Three programs are assigned for the programming section with a fourth program on vector statics given during the last part of the course (figure 2). Each part of the course consists of 15 periods, including an examination. The vector statics programming assignment attempts to demonstrate the application of the computer to the solution of an 'engineering' problem.

The science/liberal arts course, also an engineering science course, is a three-credit offering. It meets for two lectures per week and one drill period per week. Six quizzes are held approximately every other drill period (see figure 3) with a quiz average computed from the five highest quiz grades. There are five programming problems assigned. Each problem must be run on the computer. The required format for problem submission and the grading criteria is given in a hand-out -- Good Programming Demands that... (figure 4). The course grade is computed

based on the following schedule:

Quizzes 80%

Programming problems 20%

Students are encouraged to proceed at their own pace through the course material. A brief introduction to Fortran is included at the end of course, as many application programs are written in this language. The coverage of Fortran is more of a survey, although a programming assignment is required, with parallels drawn to BASIC whenever possible. Compilation and execution of a Fortran program quickly demonstrates to the student the differences between an interpreter and a compiler.

Preparation of a BASIC source program via the text editor is encouraged toward the end of the BASIC programming section of the course. For the science/liberal arts student a working knowledge of the editor is critical during his brief introduction to Fortran. For the engineering student later use of library programs require the editor when building data files. In either case, BASIC serves as a starting point for an introduction to the text editor with a 5-10 minute discussion of this subsystem incorporated in the classroom lecture over a two week period. This (editor) material is presented concurrently with BASIC programming concepts.

TUTORIAL BASIC

An integral part of a student's experience in the introductory courses is the tutorial BASIC series. This series serves as a very important adjunct to the classroom lecture. It not only drills the student on the fundamentals of the BASIC language, but through terminal access acquaints the student with log-in and log-out procedures, simple MPE commands, BASIC commands and statements, typing and keyboard layout, etc.

The tutor series presently consists of five lessons (figure 5) in BASIC. Each lesson is approximately 30-40 minutes in length. All input is character oriented and response to correct and incorrect answers generates a randomly selected typed output. The areas currently covered by the tutor series are:

introduction to BASIC

array, looping and control

functions and subroutines

strings

formatting

Topics to be added are matrix operators and file handling.

GENERAL CONSIDERATIONS

For the freshman engineering student, the brief introduction to BASIC in the second semester engineering science course is but a building block for computer applications in later courses. In the sophomore year advanced mechanics courses, circuits and a numerical math course require problem solutions via Basic programming. In the junior/senior level courses, while programming continues to be used, some emphasis is placed on using library programs (figure 6) such as COGO, ECAP and LEANS as well as programs developed strictly for departmental use. Fortran is occasionally the programming language for a few problems and Fortran is also the language for many library programs.

The science/liberal art student schedules the 3 credit engineering science course any time during his undergraduate stay at Lafayette. The course is offered each semester. After completing engineering science 24, future programming efforts are at the discretion of the departments in which he is taking courses. Most departments (e.g. psychology, physics, mathematics, education) have developed their own programs, particularly statistical routines. Library programs are also available for use (figure 6).

An advanced computer course dealing with the 3000 system (file structure and intrinsics, segmenter, etc.) is offered in alternate years. The 3 credit engineering science 24 course or its equivalent is a prerequisite. Fortran, rather than BASIC, is 'the' programming language for the course.

LAFAYETTE COLLEGE

Department of Engineering Science

E.S. 26 Spring 1975

Text: Basic Programming by Murrill and Smith

Part II: Computer Programming

Period	Topic	Pages
16	Introduction	1 - 11
17	Arithmetic, Input-Output	11 - 28
18,19	Control	29 - 52
20,21	Loops	53 - 72
22,23	Arrays	73 - 94
24	More on Input-Output	95 - 115
25,26,27	Intro. to MAT operators	123 - 127
28,29	Functions and Subroutines	116 - 122
30	Exam II	

A due date for each problem will be set by your instructor.

All programming problems must include:

(1) log-in

(2) program listing

(3) run with sample data

(4) log-off

All programs should be documented and all computer output must contain suitable headings.

Figure 1. Introductory Programming for Engineers

```
:BASIC

BASIC 3.0
>GET STATICS/SCM
>LIST
STATICS
  10 REM SOLUTION OF VECTOR STATICS PROBLEM
  20 REM LOADS ARE A, B, C
  30 REM LOADS CANNOT EXCEED 600#
  40 REM LOADS MUST BE COMPRESSIVE
  50 REM X & Z ARE TABLE DIMENSIONS
  60 PRINT
  70 PRINT " X     Z          A       B       C"
  80 FOR X=1 TO 6
  90   FOR Z=1 TO 4
 100     REM EQUILIBRIUM EQUATIONS FOLLOW
 110     C=3600/X,A=(2800-C*Z)/4,B=1200-A-C
 120     IF A<0 OR A>600 THEN 160
 130     IF B<0 OR B>600 THEN 160
 140     IF C<0 OR C>600 THEN 160
 150     PRINT USING 180;X,Z,A,B,C
 160   NEXT Z
 170 NEXT X
 180 IMAGE 2(D.DDXX),3(X6D)
 190 PRINT LIN(1),"CPU TIME =";CPU(0)
>RUN
STATICS

 X     Z          A       B       C
6.00  1.00       550      50     600
6.00  2.00       400     200     600
6.00  3.00       250     350     600
6.00  4.00       100     500     600

CPU TIME = .496

>
```

Figure 2. Vector Statics Program

LAFAYETTE COLLEGE
Department of Engineering Science
E.S. 24 Syllabus

Week of:	Topic	Reading
1/20	Introduction	pp. 1-5
1/27	simple programs	6-28
*2/3	transfer of control	29-52
2/10	loops	53-72
*2/17	arrays	73-94
2/24	input-output	95-115
*3/3	functions and subroutines	116-122
3/10	more on input-output	(95-115)
*3/17	strings	(102-104)
4/7	MAT operators	123-133
4/14	intro. to Fortran	N
*4/21	intro. to Fortran	O
4/28	Fortran	T
*5/5	applications	E
5/12	review	S

*Friday quiz scheduled for this week

Text: Basic Programming
by Murrill and Smith

Ref: HP-3000 BASIC INTERPRETER Manual
HP-3000 FORTRAN manual

Figure 3. Introductory Programming for Science/Liberal Arts Students

1. Remarks are included in the program to title the program and to explain steps.
2. All programs have labeled output.
3. Handwritten programs use valid BASIC statements (i.e., a statement number followed by a BASIC instruction. Remember that instruction words in BASIC are written in capital letters.
4. A check for a final data value is included if appropriate (i.e., the program does not end on an OUT-OF-DATA error or by typing control Y in response to an INPUT statement.
5. Counters or indexes are used where appropriate.
6. Sufficient and appropriate data is supplied in a DATA statement.
7. Functions are used rather than arithmetic statements (i.e., such things as using SQR(X) rather than X↑.5).
8. Programs to hand in include log-in, log-out and a correct <u>final listing</u> of the program.
9. A READ statement is used when data is indicated to be read and an INPUT statement used where data is indicated to be inputted.
10. The statement referred to in an IF...THEN statement is not a GO TO statement.
11. <u>All</u> arrays are DIMed.
12. GO TO's are minimized.

Grading starts @90. Points added for excellence in programming but points subtracted if any of above violated or specific program requirements not met.

Figure 4. Good Programming Demands that....

```
>RUN TUTOR.PUB
TUTOR
TUTOR IS A COLLECTION OF PROGRAMS DESIGNED TO INTRODUCE
YOU TO THE FUNDAMENTAL CONCEPTS OF THE BASIC PROGRAMMING
LANGUAGE.  BASIC (BEGINNERS ALL-PURPOSE SYMBOLIC INSTRUCTION
CODE) IS A COMPUTER PROGRAMMING LANGUAGE FOR COMPUTATIONAL
ANALYSIS, TEXT EDITING, COMPUTER AIDED INSTRUCTION, AND
MANY OTHER APPLICATIONS.
ALL PROGRAMMING LANGUAGES CONSIST OF A SET OF ORDERED
INSTRUCTIONS TO THE COMPUTER THAT PERMIT:
          ARITHMETIC CALCULATIONS
          CONTROL OF PROGRAM LOGIC
          INPUT OF DATA
          OUTPUT OF RESULTS
          SPECIFICATIONS AND FUNCTION DEFINITION.
THIS ORDERED SET OF INSTRUCTIONS IS YOUR COMPUTER PROGRAM.

THERE ARE SEVERAL LESSONS IN THIS SERIES.
THEY ARE:

     LESSON 1 - TUT01.PUB - INTRODUCTION TO THE 'BASIC' LANGUAGE.
     LESSON 2 - TUT02.PUB - ARRAYS, LOOPING, AND CONDITIONAL STATEMENTS.
     LESSON 3 - TUT03.PUB - FUNCTIONS AND SUBROUTINES.
     LESSON 4 - TUT04.PUB - STRINGS.
     LESSON 5 - TUT05.PUB - FORMATTING.

TO BEGIN YOUR TUTOR LESSONS, TYPE
          RUN TUT01.PUB
FOLLOWING THE > SYMBOL.

>RUN TUT01.PUB
TUT01

TUTOR/3000    LESSON 1

WELCOME TO THE FIRST 'BASIC' LESSON.

BEFORE WE CAN WRITE A PROGRAM WE NEED TO REVIEW THE SYMBOLS
AVAILABLE.

 /   **   -   *   +   ()


WHICH OF THE SYMBOLS IS USED FOR ADDITION?+
NICE GOING

WHICH OF THE SYMBOLS IS USED FOR SUBTRACTION?-
NOT BAD, YOU'R RIGHT

WHICH OF THE SYMBOLS IS USED FOR MULTIPLICATION?
```

Figure 5. Tutor Series

PROGRAM	PROGRAM TYPE	DESCRIPTION
BASIC	SUBSYSTEM	BASIC INTERPRETER
COBOL	SUBSYSTEM	COBOL COMPILER
COGO	FORTRAN PROGRAM	COORDINATE GEOMETRY
CPUTIME	FORTRAN SUBPROG	COMPUTES CPU TIME IN SECONDS
CUFIT	FORTRAN PROGRAM	POLYNOMIAL CURVE FITTING
CURFIT	BASIC PROGRAM	LEAST SQUARES CURVE FITTING
ECAP	FORTRAN PROGRAM	ELECTRONIC CIRCUIT ANALYSIS PROGRAM
EDITOR	SUBSYSTEM	TEXT EDITOR
FCOPY	SPL PROGRAM	FILE COPIER
FORTRAN	SUBSYSTEM	FORTRAN COMPILER
LEANS	FORTRAN PROGRAM	ANALOG SIMULATOR
LINPRO	BASIC PROGRAM	LINEAR PROGRAMMING
MULTREG	BASIC PROGRAM	MULTIPLE LINEAR REGRESSION
PLOT	FORTRAN SUBPROG	PRINTER PLOTTING ROUTINES
POLAR	BASIC PROGRAM	POLAR FUNCTION PLOTTING ROUTINE
POLRT	FORTRAN SUBPROG	REAL AND COMPLEX ROOTS OF A POLYNOMIAL
RAND	FORTRAN SUBPROG	RANDOM NUMBER GENERATOR
ROOTS	FORTRAN PROGRAM	REAL AND COMPLEX ROOTS OF A POLYNOMIAL
SIMSQ	FORTRAN SUBPROG	SOLUTION OF SIMULTANEOUS LINEAR EQUATIONS
SIMUL	BASIC PROGRAM	SOLUTION OF SIMULTANEOUS LINEAR EQUATIONS
SORT/MERGE	SPL PROGRAMS	FILE SORT/MERGE UTILITIES
SPL	SUBSYSTEM	SYSTEMS PROGRAMMING LANGUAGE COMPILER
STAR	SUBSYSTEM	STATISTICAL ANALYSIS ROUTINES
TUTOR	BASIC PROGRAM	BASIC TUTORIAL SERIES
XYPLOT	BASIC PROGRAM	X-Y FUNCTION PLOTTING ROUTINE

Figure 6. Library Programs

REFERENCES

1. BASIC Programming, Murrill and Smith, Intext (1971).

2. BASIC For Self-Study or Classroom Use, Albrecht, Finkel and Brown, Wiley (1973).

3. Basic BASIC Programming Self-Instruction Manual and Text, Peluso, Bauer and DeBruzzi, Addison-Wesley (1972).

4. LEANS (Lehigh Analog Simulator), IBM 1130 Contributed Program Library, 11.1.001.

5. IBM Electronic Circuit Analysis Program, Jensen and Lieberman, Prentice-Hall (1968).

6. Civil Engineering Coordinate Geometry (COGO) for IBM 1130 Model II, application Description, GH20-0143.