```
INCORRECT FORMAT IN 40          STRING TOO LONG IN 45
INCORRECT FORMAT IN 50          STRING TOO LONG IN 55
INCORRECT FORMAT IN 60          STRING TOO LONG IN 65
INCORRECT FORMAT IN 70          STRING TOO LONG IN 75
INCORRECT FORMAT IN 80          STRING TOO LONG IN 85
INCORRECT FORMAT IN 90          STRING TOO LONG IN 95
INCORRECT FORMAT IN 100         STRING TOO LONG IN 105
INCORRECT FORMAT IN 110         STRING TOO LONG IN 115
INCORRECT FORMAT IN 120         STRING TOO LONG IN 125
INCORRECT FORMAT IN 130         STRING TOO LONG IN 135
INCORRECT FORMAT IN 140
INCORRECT FORMAT IN 150
INCORRECT FORMAT IN 160
INCORRECT FORMAT IN 170
INCORRECT FORMAT IN 180
INCORRECT FORMAT IN 190
INCORRECT FORMAT IN 200
INCORRECT FORMAT IN 210
INCORRECT FORMAT IN 220
INCORRECT FORMAT IN 230
INCORRECT FORMAT IN 240
INCORRECT FORMAT IN 250
INCORRECT FORMAT IN 260
INCORRECT FORMAT IN 270
INCORRECT FORMAT IN 280
INCORRECT FORMAT IN 290
INCORRECT FORMAT IN 300
INCORRECT FORMAT IN 310
INCORRECT FORMAT IN 320        STRING TOO LONG IN 325
INCORRECT FORMAT IN 330        STRING TOO LONG IN 335
INCORRECT FORMAT IN 340        STRING TOO LONG IN 345
INCORRECT FORMAT IN 350        STRING TOO LONG IN 355
INCORRECT FORMAT IN 360        STRING TOO LONG IN 365
INCORRECT FORMAT IN 370        STRING TOO LONG IN 375
INCORRECT FORMAT IN 380        STRING TOO LONG IN 385
INCORRECT FORMAT IN 390        STRING TOO LONG IN 395
INCORRECT FORMAT IN 400        STRING TOO LONG IN 405
INCORRECT FORMAT IN 410        STRING TOO LONG IN 415
INCORRECT FORMAT IN 420        STRING TOO LONG IN 425
INCORRECT FORMAT IN 430        STRING TOO LONG IN 435
INCORRECT FORMAT IN 440        STRING TOO LONG IN 445
INCORRECT FORMAT IN 450        STRING TOO LONG IN 455
INCORRECT FORMAT IN 460        STRING TOO LONG IN 465
INCORRECT FORMAT IN 470        STRING TOO LONG IN 475
INCORRECT FORMAT IN 480        STRING TOO LONG IN 485
INCORRECT FORMAT IN 490        STRING TOO LONG IN 495
INCORRECT FORMAT IN 500        STRING .\XSTOP.
READY.
```

PROGRAMS IN BASIC

Steve Alcorn John Stuppy

cover design by Steve Alcorn

PROGRAMS IN BASIC

Term Project

Steve Alcorn John Stuppy
In Conjunction With The
Math Dept. Tape Library
Collected And Compiled By
The Same

** * **

TOPIC OUTLINE

TOPIC: Computer programs involving, unucual, interesting, or useful functions compiled by the Geometry/Algebra 1st period class and John Stuppy & Steve Alcorn.

INSTRUCTOR: Mr. George B. Roberts.

FACILITIES: Harvard School, North Hollywood, Calif., User Number K30100.

GRADE: Ninth.

COURSE: Geometry/Algebra.

CONTENTS: Introduction, Description of Program Functions, Programs, Program Descriptions.

PROGRAMS IN BASIC

Steve Alcorn John Stuppy

Unit Length: Geometry/Algebra: 6 Mo., 5 Days per Week.
Computer Programming: 6 Weeks, 5 Days per Week.

Text: Basic Manual, Mark 1 Time-Sharing Service, Reference Manual, G.E.

Prerequisites: Algebra Geometry, 2 semesters

Purpose : To present a comprehensive but brief series of operations, their practical applications, and most advantageous use in Basic.

The following is a collection of programs of Steve Alcorn and John Stuppy. In addition to their own and there classmates, also included are samples from various texts, which are listed in the credits.

TABLE OF CONTENTS

INTRODUCTION

Why a High School computer course? Busy work? Practice for becoming a computer programmer? No, the age of computers is dawning. Whether or not you are now planning to become a computer programmer, it is most evident that sometime in the future, you will have some interfacing with computers. It was the purpose of the computer course, taken in conjunction with the Algebra-Geometry class, to introduce us to this new and important field.

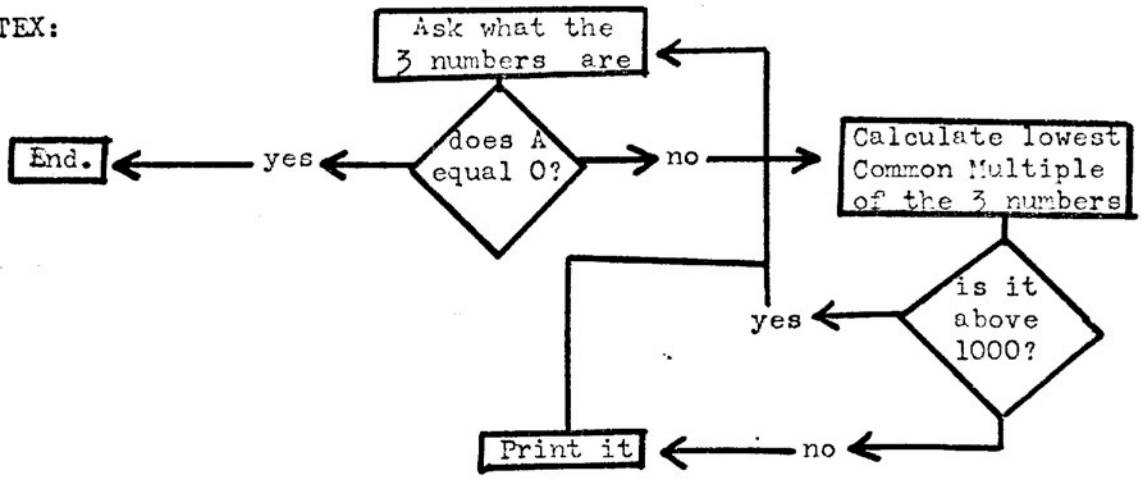
For the most part, the programs we did were parallel to our study of Algebra and Geometry. In other words, we learned something dealing with Algebra or Geometry and then composed a computer program using this new information. As time went on, we increased our knowledge of BASIC functions and operations, and set out to use our knowledge on "bigger and better" programs. Our interests branched out, and we found that a computer was a lot more than an expensive calculator. It was useful in practically every course we took. It could translate and conjugate verbs in foreign languages, make histograms and sort data in Science, and of course solve any mathematical function desired. The computer could even be programed to play tic-tac-toe. One boy found it useful to go through the next day's races at the race track and have the computer "pick a winner".

Before you start flipping the pages to get to the programs mentioned above, I feel I must explain something: The computer is not a toy. It must be operated skillfully, and you must know what you are doing. All the programs we ran were generally related to our study of mathematics, or our other studies. The fact is that each program increased our understanding of BASIC language and computer programming. In other words,a program for finding the numbers from 1 to 100 that have a 7 in them (like 72), or are divisible by 7 (like 21), is just as useful as a program for calculating pi to the millionth decimal place, in that with each program you write, you become more proficient in computer programing. Though the

first program mentioned is not nearly as useful as the second as far as answers are concerned, it, like all other programs, is a challenge to write.

In the beginning of the year when we first began to learn about computers we were taught about flow charts. A flow chart is a diagram of the operation of a computer program. Below is an example:

MULTEX:



(note: this program is located somewhere in the book. it finds the L.C.M. of 3 numbers.)

As you can see, a flow chart is a very logical, step-by-step solving of a problem. It is now that I wish to make my final point: A computer program is one big flow chart, not in the sense that you have questions in boxes with confusing arrows all over the place, but in the sense that the calculations and functions in a computer program fall into a logical order: You must know where you are going. And by writing programs, one may develop a logical mind, which is useful in all phases of life.

PROGRAMS IN BASIC

It is assumed, of course, that before opening this book, the reader has a firm understanding of Basic Language, or , more primarily, knows it's applications and limitations. However, It is not safe to assume that the reader is aware of all of the programing statements which will be discussed. Therefore we feel it necessary to include a listing of them with a brief description of their uses below:

1. FOR/NEXT LOOP--This Loop sets a value range for the variable in the 'for' statement and then, in the 'next' statement returned the program flow to the 'for' loop where it is incremented according to the 'step'. (If not otherwise specified, the step is one.) As soon as the variable reaches the upper value of the predefined range, the program flow immediately drops to the statement directly below 'next'.

```
100 FOR A = 1 TO 25 STEP 5
   (lines 110 - 150)
160 NEXT A
```

or

```
100 FOR A = 1 TO 25
   (lines 110 - 150)
160 NEXT A
```

2. IF/THEN STEP--If variable equals (less than, etc.) preset value program flow skips to line set by 'then'.

```
100 IF A = 0 THEN 999
   (lines 110 - 990)
999 END
```

3. GOTO--as anIF/THEN, except no proviso. Automatically shifts to designated line.

```
100 GOTO 999
   (lines 110 - 990)
999 END
```

4. DIM--Strings can be set up as one-dimensional arrays only. Examples:

```
100 DIM A(5),C$(20),A$(12),D(10,5)
200 DIM R$(35)
300 DIM M$(15),B$(15)
```

5. LET--Sets variable equal to variable, number, or calculation
Examples:

```
100 LET A = B
200 LET A = 3
300 LET A = 2/R*87-45
```

6. ALPHANUMERIC DATA--Letting a variable represent letters, words, or phrases rather than numbers. (note: cannot be longer than 15 characters)

```
100 LET A$ = "HOW ARE YOU";
110 INPUT A$
120 IF A$ = "FINE" THEN 999
```

7. READ/DATA--Although most if not all of the programs in this book contain input statements, READ/DATA statements may be used instead. (Note: DATA may be anywhere before 'END'.)

```
100 READ A,B,A$
990 DATA 1,4,"HELLO"
```

8. PRINT--In addition to normal print statements, 'PRINT' and a blank line skips a space and " " leaves a space.

```
100 PRINT "   ";"   ";"HELLO","   ";B,"   ";4/5*F
110 PRINT
120 PRINT "GOODBYE"
130 PRINT A$;
```

9. TAB--Used in print statement, it leaves a space of prescribed length.

```
100 PRINT X; TAB(10);Y;
```

10. INT--Finds the integer part of numbers. INT(7.864) is 7

11. RND--Generates random numbers. Check a basic manual for further detail

12. DEF--Defines a function

```
130 DEF FNF(X)= X-Y/(Y-4)
```

13. GOSUB/RETURN--Goes to sub routine and returns

14. STOP--Immediately goes to end

15. REM--Remark to yourself, ignored by computer. 100 REM STEVE ALCORN