COMPUTER PROGRAMMING EXAMPLES FOR CHEMICAL ENGINEERS

GEORGE ROSS

COMPUTER-AIDED CHEMICAL ENGINEERING, 3

ELSEVIER

COMPUTER-AIDED CHEMICAL ENGINEERING, 3

COMPUTER PROGRAMMING EXAMPLES FOR CHEMICAL ENGINEERS

GEORGE ROSS

Department of Manufacturing Engineering, Swinburne Institute of Technology, Hawthorn, Victoria, Australia

ELSEVIER
Amsterdam –Oxford – New York – Tokyo 1987

ELSEVIER SCIENCE PUBLISHERS B.V.
Sara Burgerhartstraat 25
P.O. Box 211, 1000 AE Amsterdam, The Netherlands

Distributors for the United States and Canada:

ELSEVIER SCIENCE PUBLISHING COMPANY INC.
52, Vanderbilt Avenue
New York, NY 10017, U.S.A.

ISBN 0-444-42836-4 (Vol. 3) (hardbound)
ISBN 0-444-42837-2 (Vol. 3) (paperback)
ISBN 0-444-42482-2 (Series)

Printed in The Netherlands

INTRODUCTION

Purpose

In the chemical engineering design field, computers are used in the preparation of heat and mass balances, the selection of optimum flow paths, the design of equipment items such as heat exchangers and distillation columns, and for the preparation of flow diagrams, layouts and mechanical drawings. Laboratory applications include a range of chemical analyses, determinations of particle size ranges, etc. whilst on the plant, computers and microprocessors are rapidly replacing conventional analog measurement and control devices.

How is the average chemical engineer equipped to handle these new developments? For many older engineers, the computer came along only as an afterthought in their studies, if it came at all. Current students of course, are much better equipped and receive tuition in programming, so that they are able to write programs in one or more of the common languages such as Basic, Fortran, Pascal, APL & C, but even they have been overtaken by the recent enormous increase in both power and availability of micro or desk-top computers such as the IBM PC. They are unlikely also, before completing their tertiary studies, to gain much experience writing programs related to their own engineering discipline.

Outside of teaching and research, many computers exclusively use software which has been developed by the computer manufacturers or other specialist companies. Such software packages are produced by large teams of programmers, numbering perhaps in the hundreds, working together for months under the direction of a co-ordinator. The average user of such a package is unlikely to be able to understand the program, even if he has access to it.

On entering employment, the young graduate may well find that where computers are used, such software packages are used with them. Many of these can now be run on a microcomputer equipped with a hard disc. In these circumstances, the user finds his function limited to the level of inputting data to the machine and making decisions with regard to its output, often without regard for the theoretical principles on which the software is based. I think this is a potentially dangerous situation, ultimately tending to degrade the profession, and restricting high level skills to the few.

The purpose of this book is to encourage the engineer to apply the skills in programming which he has learnt to the solution of problems in engineering. Programming such solutions is not difficult provided the programmer has a good understanding of the principles to be applied. Any gaps in this understanding will soon be uncovered! I am sure that an ability to write such programs, and the confidence which goes with it, is invaluable to the engineer working for the small company, which cannot justify the expense of the sophisticated software packages. For the engineer who does work with such packages, I hope this book will provide some insights into the logic which they employ, and encourage him also to develop his own programs.

I have selected a number of topics of interest to chemical engineers, a separate chapter being devoted to each. Each chapter presents the theoretical principles in summary form, then takes the reader through one or more manual calculations, in detail. Then a computer program is presented to perform the same task, each program including a detailed description, and being preceded by a logic flowchart.

Programming Language

The programs in this book are all written in BASIC.

I have chosen to do this for several reasons. It is the most widely used programming language and is acknowledged to be the easiest to learn, and it was the first language adopted for use on microcomputers, with which it is still widely used. Despite certain limitations with regard to speed and storage, Basic in its more developed forms is quite powerful enough for the average user.

The programs are prepared on IBM PC and PC/XT machines employing Microsoft Basic. The diskette which may be purchased with this book will of course run on these machines and on other IBM - compatible machines. The form of Basic used is in conformity with A.N.S.I. (American National Standards Institute) and so is suitable for many other Basic compilers.

In certain cases the diskette may be unsuitable; this occurs where the byte size employed in a particular machine is different from that used on the IBM machine. In such a case, unless a translator is available, there is no recourse but to type the programs in at the keyboard. Complete program listings are provided which enable this to be done.

Learning to Program

It may be that you have not yet learned to program. Don't be put off, it is not difficult. The first step is to obtain a Basic Manual appropriate to the machine you will be using. Next obtain a teaching text, of which a number are available (1) (2) (3) (4) (5). With these in hand you should be able to follow all the programs given in this book, and to extend them.

If you wish to progress to more advanced topics, there are also texts which will help you to do this (6) (7).

Logic Flowcharts

These are intended to make it easier to follow the program logic. They display the principal features of each program only, and employ symbols now standardised by A.N.S.I. and I.S.O. (International Organisation for Standardisation). The symbols used in this book are tabulated below, with brief summaries of their functions. For more information on flowcharts consult references already cited (2) (3).

Mathematics

The jobs which the chemical engineer is likely to use the computer for fall into the following categories:

1. Storage of data (for example, physical & thermodynamic properties).
2. Interpretation of data (that is, finding a mathematical relationship to fit the data).
3. Solution of problems involving stagewise processes (distillation, liquid extraction, evaporation, etc), treated as a series of lumped parameter systems.
4. Solution of problems in heat, mass and momentum transfer possibly involving transients. These are usually distributed parameter systems, but may be modelled as series of lumped parameter systems by the methods of finite differences (also by finite element methods).

The techniques employed to solve these problems are simply:

- Standard analytical mathematical methods
- Statistical methods
- Finite difference methods

Usually, large numbers of simultaneous equations are generated and have to be solved. This can be done either by Iteration, or by Matrix Algebra.

The chemical engineer who wishes to do his own programming then has to have some facility with each of the above. However, the necessary skills can be developed as the need arises.

Warnings

Commercial software includes a great deal of programming whose only function is to protect the software and to avoid user mistakes. Very little of such measures is included with the programs in this book; it makes understanding of the programs very much harder. If for instance, the program calls for values of composition in weight fraction, then the user must ensure that the sum of these values equals unity. If they do not, then errors will arise. So it is necessary to be careful in your working. If you don't like this situation then you can obtain excellent programming practice by incorporating your own safety precautions!

It is recommended that you purchase the program diskette if possible. It will save you many hours of typing and then checking for your mistakes. If you do possess the diskette, make a copy of it immediately, and use this copy when running or working on the programs. The original diskette should be retained as a master copy and only used to restore your working diskette if this has been corrupted.

The asterisk * is used in the text to indicate multiplication, in order to avoid confusion with the letter x.

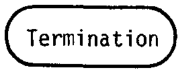
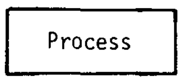
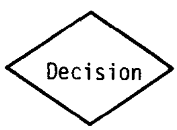
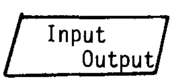
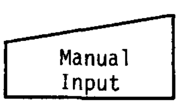
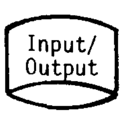
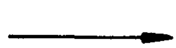
This book has been written to encourage chemical engineers to develop their programming skills. I hope you find it helpful.

G. Ross
Swinburne Institute of Technology
Melbourne 1987

REFERENCES - INTRODUCTION

1 W. Amsbury, Structured Basic and Beyond, Computer Science Press, Rockville MD, U.S.A. 1980
2 H.A. Moriber, Structured Basic Programming, Charles E. Merrill Publishing Co. Columbus, OH, U.S.A., 1984
3 R.G. Thompson, Basic - A Modular Approach, Charles E. Merrill Publishing Co. Columbus, OH, U.S.A., 1985
4 L. & J. Rosenblatt, Simplified Basic Programming: with Companion Problems, Addison - Wesley Publishing Co., Reading Mass, U.S.A. 1973
5 S.L. Marateck, Basic, Academic Press, New York, U.S.A., 1975
6 I. Stewart, R. Jones, Machine Code and Better Basic, Shiva Publishing, Nantwich, U.K. 1982
7 M.L. Lesser, Using Microsoft Compiled Basic, McGraw-Hill Book Co., New York, U.S.A., 1985

FLOW CHART SYMBOLS

Symbol	Meaning or Application
	This symbol indicates a terminal point, such as the beginning or end of a program or subroutine.
	This stands for any function or group of functions causing changes in the information flow, for example, it could be a sequence of algorithms used in a calculation.
	Statements causing branching, such as IF ... THEN or FOR ... NEXT. An abbreviated expression for the decision is placed inside the symbol with a ? below it.
	For example, input from a data statement, output to CRT or printer.
	Input from the keyboard.
	Input/Output from/to a magnetic disc or similar device.
	Connector, used for convenience in laying out the flowchart, it has no counterpart in the actual program.
	A line with an arrowhead is used to link symbols; the direction of the arrow indicates the sequence of operations and the direction of the data flow.
	The dotted line has no counterpart in the actual program; its purpose is to indicate the symbols referred to in a comment or annotation appended to the flowchart.

CONTENTS

Chapter 1

FLOWSHEETING (PROCESS SIMULATION)

Preparation of flowsheets is an important part of the work of the design chemical engineer. The flowsheet is the bridge between design calculation and plant hardware. Choice of optimum number of stages of extraction, or of heat exchange, depends upon material and energy balance considerations, and may be regarded as part of the flowsheeting process.

Computer flowsheeting is an attempt to assemble many chemical engineering design functions into one package (1) (2). Software packages of considerable complexity are now available (3) (4). Usually the simulation is limited to steady state conditions, thus greatly reducing the complexity of the problems to be dealt with.

A program for flowsheeting purposes obviously cannot be written as a simple linear program. Instead it must consist of a control program and a number of subroutines. A different subroutine is required for each step or operation in the process. A good many of these would be required, and the list might include "mix", "split", "compress", "expand", "flash", "distil", "pump","heat exchange", "extract", "absorb", "settle", "react", etc.

The control program would operate in such a way that these subroutines could be called upon by the designer as required, interactively at the terminal. The designer would assemble the complete flowsheet by appropriate connections between the subroutines.

Each subroutine should perform the function of calculating material and energy balances for the process step which it represents. In order to do this it would generate (or call from other subroutines) data on physical properties of all the fluids and solids concerned with that process step (4). In addition, it might be necessary in some cases, such as a multi-component distillation, for further design calculations to be made within the subroutine.

Since capital and operating costs affect the process design, it might be necessary also to include further subroutines to handle cost calculations (4).

As the assembly of the flowsheet continues, the designer may wish to incorporate corrections and improvements; he may wish also from time to time to observe the effects of variation in process parameters. The program should include means to enable this to be done; it should not be necessary to start again from the beginning each time a modification has to be incorporated.

Obviously, a program to carry out all the functions mentioned above has become indeed a "software package". It will have developed gradually by the accumulation of new subroutines and by the extension of existing ones. Reference to advertisements in the technical press shows how software companies are constantly updating their material.

Typing in of the data and interpretation of the output will be difficult and confusing for a flowsheet of any complexity. Consequently graphical representations of the flowsheet and display of values would be extremely advantageous. However, it is not an essential part of the program, and the example in this chapter will show how a flowsheet program can be written in BASIC without the use of graphic output.

The program to be described involves no mathematics other than algebra; only simple process steps are included, but a more complex form of the program might incorporate material from the other chapters of this book. Writing of the program depends of course on the programmers ability to model the process (5).

The growth of the program will be described step by step so that the reader can follow the development of the logic.

STEP ONE

The beginning program must contain the minimum of necessary ingredients to operate in the intended manner, namely:

- the control program;
- a subroutine to handle data inputs and commands from the designer;
- a subroutine to simulate a simple process;
- a subroutine to handle the output of calculated values.

The Process Subroutine (MIX)

The operation of mixing of several process streams will be modelled. Given the flowrates, compositions and enthalpies of n entering streams, the flowrate, enthalpy and composition of the mixed exit stream should be calculated (see Figure 1.1). For the sake of simplicity the algorithms written here will be limited to the calculation of flowrate and composition (not enthalpy).

If we have J entering streams, then the flowrate leaving

$$F_{J+1} = F_1 + F_2 + \ldots F_J \qquad (1.1)$$

If there are K components present in each stream then the composition of the exit stream,

$$x_{J+1,K} = \frac{(x_{1,K} * F_1 + x_{2,K} * F_2 + \ldots . x_{J,K} * F_J)}{F_{J+1}} \qquad (1.2)$$

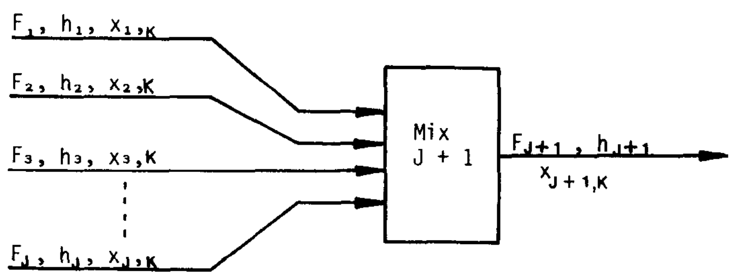
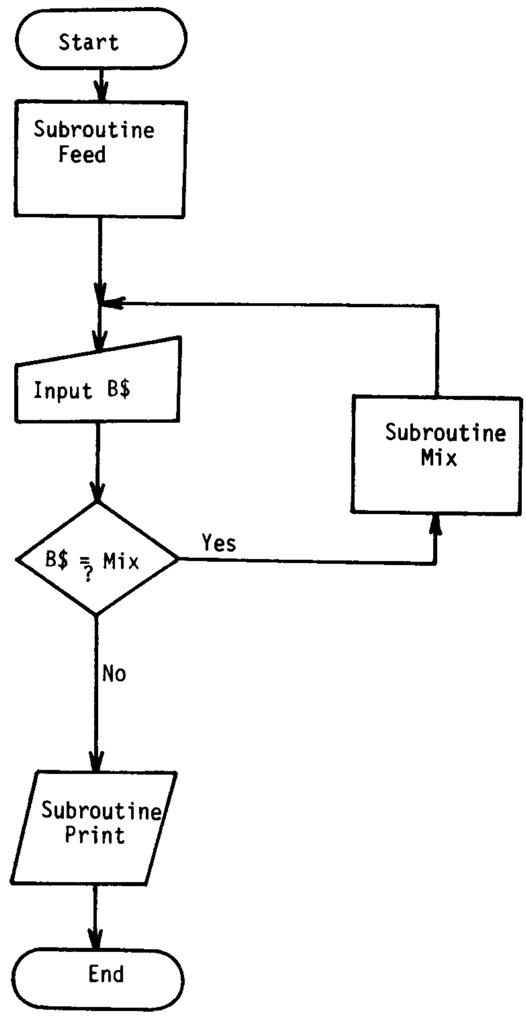


Fig. 1.1. Mixing of Streams.

This calculation should be performed for each component, i.e. for each value of the subscript K. In terms of the program, we can store all these values of composition in a two dimensional matrix X(J,K) where the first dimension refers to the number of the entering or leaving stream, and the second dimension refers to the components. We can then proceed to write a simple subroutine embodying the above two equations.

The program FSHT1 which follows, embodies this subroutine and the steps outlined above.

FSHT1. BAS


Start
Subroutine Feed
Input B$
B$ =? Mix
Yes
Subroutine Mix
No
Subroutine Print
End

```
10   REM *****************************************************
20   REM - PROGRAM FSHT1.BAS    FIRST STEP IN
30   REM - DEVELOPMENT OF A FLOWSHEET PROGRAM
40   REM - PROGRAM NOMENCLATURE:
50   REM - A$(J)     -     Names of subroutines and box numbers
60   REM - B$        -     Names of subroutines
70   REM - B(J)      -     Numbers of the streams entering
80   REM                   a subroutine box
90   REM - C$(J)     -     Names of components
100 REM - E1        -     Sum of values of F(J)
110 REM - E2        -     Sum of values of the product F*x
120 REM - E3        -     Sum of values of weight fractions
130 REM - F(J)      -     Flowrates
140 REM - N1        -     Number of components
150 REM - N2        -     Number of "boxes"
160 REM - N3        -     Number of flows to be mixed
170 REM - X(J<K)    -     Composition as weight fraction of
180 REM                   component K in stream from Box J
190 REM - Y(J<K)    -     Composition as weight fraction of
200 REM                   component K in stream from Box J
210 REM - PROGRAM DESCRIPTION
220 REM - LINES 1000,1010 Alphanumeric arrays are declared,
230 REM   and the arbitrary value of 20 allocated for the
240 REM - largest likely number of operations or "boxes"
250 REM - on the flowsheet.
260 REM - LINES 1020-1110 The control program;subroutine
270 REM - "Feed" is called first;then the designer is
280 REM - enabled to call subroutine "Mix" as required;
290 REM - finally subroutine "Print" is called.
300 REM - LINES 1130-1390 FEED subroutine;
310 REM - first, the number of components present is
320 REM - is established and these are named
330 REM - (lines 1180-1220).   Next, each entering stream
340 REM - is given a number or identifier referred to as
350 REM - a "Box Number", which the designer must keep a
360 REM - record of on his flow diagram.   As each stream is
370 REM - numbered it is also given the name "Feed", and
380 REM - values of flowrate and weight fraction of each
390 REM - component are entered (lines 1230-1380)
400 REM - LINES 1420-1650 MIX subroutine;each time this
410 REM - subroutine is called a box number is allocated
420 REM - and the box labelled "Mix" (lines 1430-1450).  The
430 REM - designer then enters the number of streams to be
440 REM - mixed and the boxes from which they arise(lines
450 REM - (1460-1490).   The algorithms previously written
460 REM - (equations 1.1 and 1.2) are then employed to
470 REM - calculate flowrate and composition of the mixed
480 REM - stream leaving(lines 1500-1640).
490 REM - LINES 1670-1780 PRINT subroutine; the name and
500 REM - number of each box is printed out with values of
510 REM - flowrate and weight fraction of each component.
520 REM - ***************************************************
1000 DIM A$(20),C$(10)
1010 DIM B(20),F(20),X(20,10),Y(20,10)
1020 GOSUB 1130
1030 PRINT "ENTER MIX,OR END"
1040 INPUT B$
1050 IF B$="MIX" THEN 1080
1060 IF B$="END" THEN 1100
1070 GOTO 1030
1080 GOSUB 1400
```

```
1090 GOTO 1030
1100 GOSUB 1670
1110 GOTO 1790
1120 REM - ***********************************************
1130 REM - FEED SUBROUTINE FOR FLOWRATES &
1140 REM - COMPOSITIONS OF ENTERING STREAMS
1150 PRINT "FLOWRATES & COMPOSITIONS FOR FLOWS"
1160 PRINT "ENTERING THE SYSTEM;ENTER THE NUMBER"
1170 PRINT "OF COMPONENTS & STICK TO THIS NUMBER"
1180 INPUT "FOR ALL STREAMS";N1
1190 FOR K=1 TO N1
1200 PRINT "NAME OF COMPONENT";K;
1210 INPUT C$(K)
1220 NEXT K
1230 PRINT "NUMBER OF STREAMS"
1240 INPUT N2
1250 FOR J=1 TO N2
1260 A$(J)="FEED"
1270 PRINT "FLOW";J;":"
1280 PRINT "FLOWRATE";
1290 INPUT F(J)
1300 IF F(J)=0 THEN 1390
1310 E3=0
1320 FOR K=1 TO N1-1
1330 PRINT "WT. FRACTION OF ";C$(K);
1340 INPUT X(J,K)
1350 E3=E3+X(J,K)
1360 NEXT K
1370 X(J,N1)=1-E3
1380 NEXT J
1390 RETURN
1400 REM - ***********************************************
1410 REM - MIX SUBROUTINE, SIMULATES MIXING OF
1420 REM - STREAMS CONTAINING UP TO 10 COMPONENTS
1430 N2=N2+1
1440 PRINT "THIS IS MIX SUBROUTINE,BOX NUMBER";N2
1450 A$(N2)="MIX"
1460 INPUT "NUMBER OF STREAMS TO BE MIXED";N3
1470 FOR J=1 TO N3
1480 INPUT "BOX NUMBER FROM WHICH STREAM COMES";B(J)
1490 NEXT J
1500 E1=0
1510 FOR J=1 TO N3
1520 E1=E1+F(B(J))
1530 NEXT J
1540 FOR K=1 TO N1
1550 E2=0
1560 FOR J=1 TO N3
1570 E2=E2+F(B(J))*X(B(J),K)
1580 NEXT J
1590 Y(N2,K)=E2/E1
1600 NEXT K
1610 F(N2)=E1
1620 FOR K=1 TO N1
1630 X(N2,K)=Y(N2,K)
1640 NEXT K
1650 RETURN
1660 REM - ***********************************************
1670 REM - PRINT SUBROUTINE
1680 FOR J=1 TO N2
1690 PRINT A$(J)
```