CHAPTER 1

# INTRODUCTION TO PROGRAMMING

## INTRODUCTION

A computer program is a sequential set of instructions written in a computer language that is used to direct the computer to perform a specific task of computation.

Observe that the definition demands that any set of instructions must be such that the tasks will usually be performed sequentially unless directed otherwise. Each instruction in the set will express a unit of work that a computer language can support. In general, high level languages, also known as 3GLs, support one human activity at a time. For example, if a computational task involves the determination of the average of three numbers, then it will require at least three human activities, *viz*., getting the numbers, obtaining the sum of the numbers, and then obtaining the average. The process will therefore require three instructions in a computer language. However, it can be done using two instructions, also: first by obtaining the numbers and second by obtaining the sum and the average.

The objective of programming is to solve problems using computers quickly and accurately.

## FLOWCHARTING AND ALGORITHMS

A problem is something the result of which is not readily available. A set of steps involving arithmetic computation and/or logical manipulation is required to obtain the desired result. There is a law called the *law of equifinality* that states that the same goal can be achieved through different courses of action and a variety of paths, so the same result can be derived in a number of ways. For example, consider the task of sending a message to one of your friends. There are many ways in which this can be done. First, you can convey the message over the phone if your friend possesses a phone. Second, you can send it by post. Third, you can send it through a courier service. If the message is urgent, then you can try to use the quickest means for sending it. If it is not urgent, then you will choose to send it in the least expensive but most reliable way of doing it. Depending upon the urgency, you will decide the most effective way of doing it. This most effective way is called the *optimum way*. The different ways of solving a problem are called *solution strategies*. The optimum way of solving a problem to get the desired result can be achieved by analyzing different strategies for the solution and then selecting the way that can yield the result in the least time using the minimum amount of resources. The selection process will depend on the efficiency of the person and his/her understanding of the problem. He/she must also be familiar with different problem-solving techniques. Determining the set of steps required to solve a given problem is an art. It shows how well a person can arrange a set of steps so that others can follow it. A type of analysis called *task analysis* is required to reach the solution from a problem definition that states what is to be achieved.

A set of steps that generates a finite sequence of elementary computational operations leading to the solution of a given problem is called an *algorithm*. An algorithm may be too verbose to follow. The textual description of an algorithm may not be understood quickly and easily. This is why a pictorial representation may be used as a substitute for an algorithm. Such a pictorial representation is called a *flowchart*. Formally speaking, a flowchart is a diagrammatic representation of the steps of an algorithm. In a flowchart, boxes of different shapes are used to denote different types of operations. These boxes are then connected by lines with arrows denoting the flow or direction to which one should proceed to know the next step. The connecting lines are known as *flow lines*. Flowcharts may be classified into two categories:

**(i)** Program Flowchart **(ii)** System Flowchart

*Program flowcharts* act like mirrors of computer programs in terms of flowcharting symbols. They contain the steps of solving a problem unit for a specific result.

*System flowcharts* contain the solutions of many problem units together that are closely related to each other and interact with each other to achieve a goal. We will first focus on program flowcharts.

A *program flowchart* is an extremely useful tool in program development. First, any error or omission can be more easily detected from a program flowchart than it can be from a program because a program flowchart is a pictorial representation of the logic of a program. Second, a program flowchart can be followed easily and quickly. Third, it serves as a type of documentation, which may be of great help if the need for program modification arises in future.

The following five rules should be followed while creating program flowcharts.

- Only the standard symbols should be used in program flowcharts.
- The program logic should depict the flow from top to bottom and from left to right.
- Each symbol used in a program flowchart should contain only one entry point and one exit point, with the exception of the decision symbol. This is known as the *single rule*.
- The operations shown within a symbol of a program flowchart should be expressed independently of any particular programming language.
- All decision branches should be well-labeled.

The following are the standard symbols used in program flowcharts:

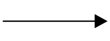
*Terminal*: used to show the beginning and end of a set of computer-related processes

*Input/Output*: used to show any input/output operation

*Computer processing*: used to show any processing performed by a computer system

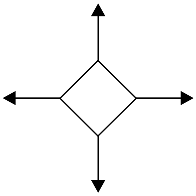
*Predefined processing*: used to indicate any process not specially defined in the flowchart

*Comment*: used to write any explanatory statement required to clarify something

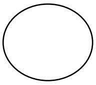


*Flow line*: used to connect the symbols

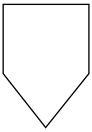
*Document Input/Output*: used when input comes from a document and output goes to a document.



*Decision*: used to show any point in the process where a decision must be made to determine further action



*On-page connector*: used to connect parts of a flowchart continued on the same page

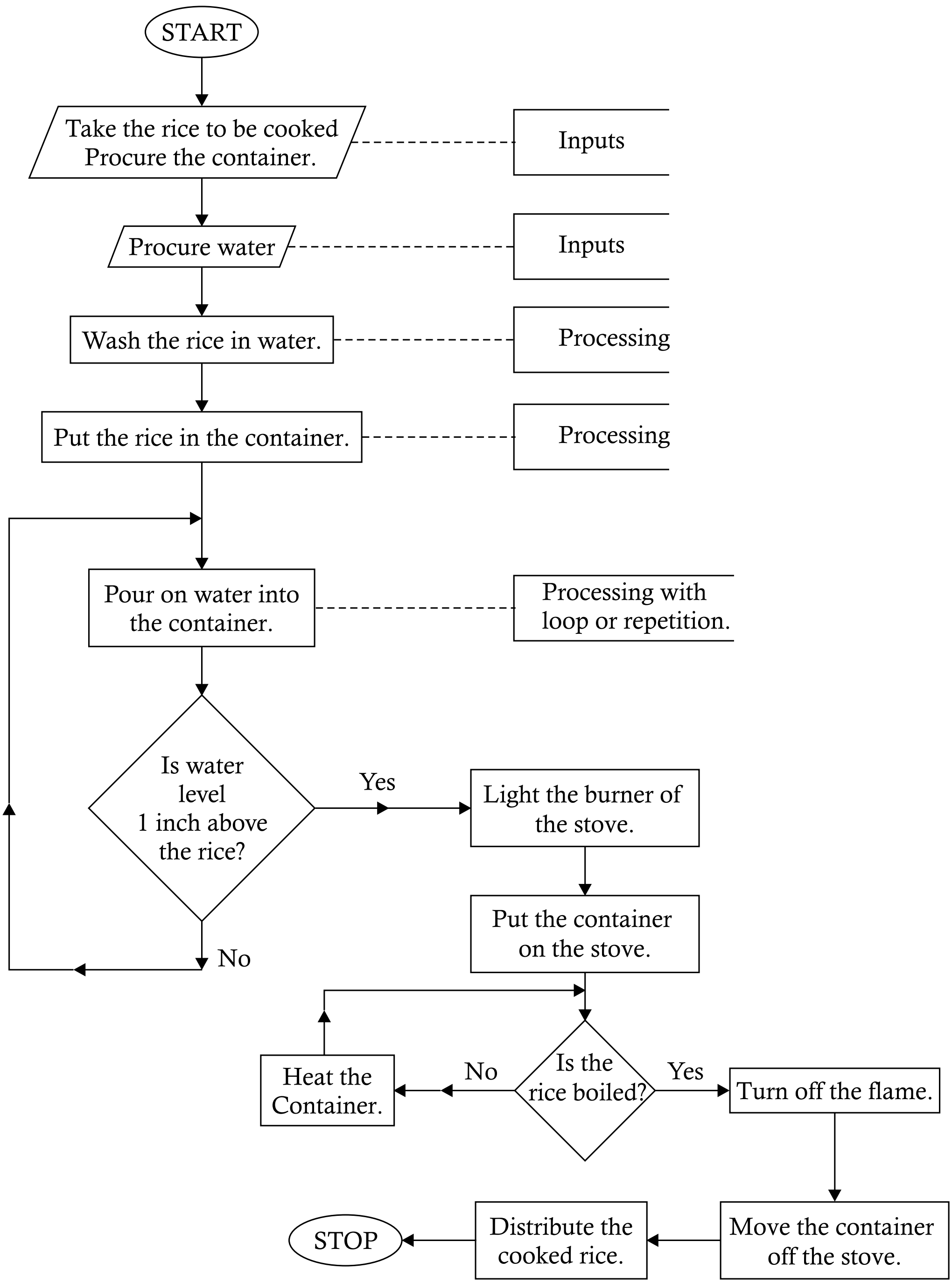


*Off-page connector*: used to connect parts of a flowchart continued to separate pages

Flowcharts can be used to show the sequence of steps for doing any job. A set of simple operations involving accepting inputs, performing arithmetic operation on the inputs, and showing them to the users demonstrate the *sequence logic structure* of a program. The following flowchart shows the steps in cooking rice and then utilizing the cooked rice.

The algorithm for the flowchart about cooking rice is as follows:

**Step 1.** Take the rice to be cooked.
**Step 2.** Procure the container.
**Step 3.** Procure the water.
**Step 4.** Wash the rice in the water.
**Step 5.** Put the rice into the container.
**Step 6.** Pour water into the container.
**Step 7.** IF WATER LEVEL = 1 INCH ABOVE THE RICE
    THEN GOTO STEP 8
    ELSE GOTO STEP 6
  ENDIF


START
Take the rice to be cooked
Procure the container.
Inputs
Procure water
Inputs
Wash the rice in water.
Processing
Put the rice in the container.
Processing
Pour on water into the container.
Processing with loop or repetition.
Is water level 1 inch above the rice?
Yes
No
Light the burner of the stove.
Put the container on the stove.
Is the rice boiled?
No
Yes
Heat the Container.
Turn off the flame.
Move the container off the stove.
Distribute the cooked rice.
STOP

**Step 8.** Light the burner on the stove.
**Step 9.** IF THE RICE IS BOILED
    THEN GOTO STEP 12
    ELSE GOTO STEP 10
  ENDIF
**Step 10.** Heat the container.
**Step 11.** Go to step 9.
**Step 12.** Turn off the flame.
**Step 13.** Move the container off the stove.
**Step 14.** Distribute the cooked rice.
**Step 15.** STOP.

The main purpose of flowcharting is to discover/invent the sequence of steps for showing the solution of a problem through arithmetic and/or logical manipulations for which we can instruct computers. The problems for flowcharting and algorithm development that we will consider here are based primarily on computational procedures.

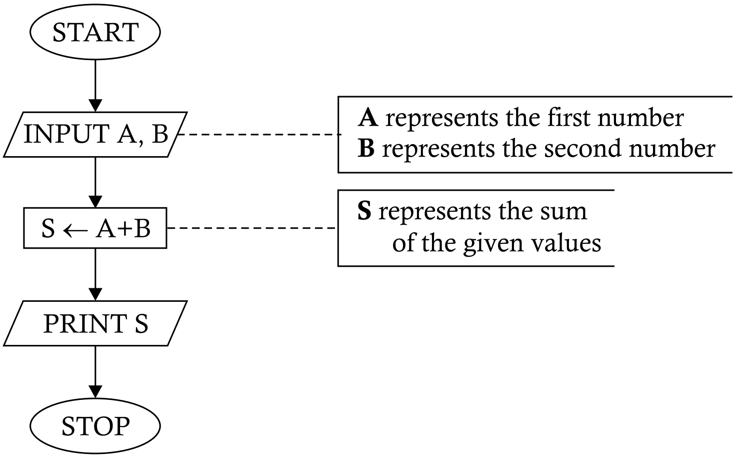
We now consider different problem definitions, followed by the task analysis, and then the desired flowchart. We denote the assignment operation using an arrow sign. The direction of the arrow implies the destination of the assignment. For example, "A ← B" means that the value contained in B is assigned to A. This, however, does not mean that the value of B is lost in A; it implies that the value contained in B is copied into A so that A and B contain the same thing. We use the symbol * or *x* to indicate a multiplication operation.

Let us consider a problem, the goal of which is to construct a flowchart to show the procedure to obtain the sum of two given numbers.

This is a very simple task. To solve the problem, we require two numbers as inputs. The numbers can then be added together to derive the sum. Observe that as a user of the procedure, you can provide any two numbers. As we wish to construct a procedure, we should not specify any arbitrary pair of numbers for the procedure. It is more convenient if we denote the input values symbolically. Symbols represent the given numbers. A similar symbol can be used to represent the sum. Another concept used during programming is that of *containers* called *variables*. The symbols for representing input data values or the output results may be treated as the containers of the values' input or output. Whatever they are, the data values are the contents of the variables. Variables are symbolic representations of containers for holding data or information. We follow the convention that a single word consisting of one to any number of characters can be used as the name of a variable. A *variable* is actually a named collection of one or more memory locations of a computer treated as a single container. Its content may vary depending on a user's operation. The following discussion explains the following flowchart of the desired procedure.

The program logic structure illustrated in the flowcharts of this chapter is the sequence logic structure.

**Problem 1.1.** *Draw a flowchart to show how the sum of two numbers can be obtained.*

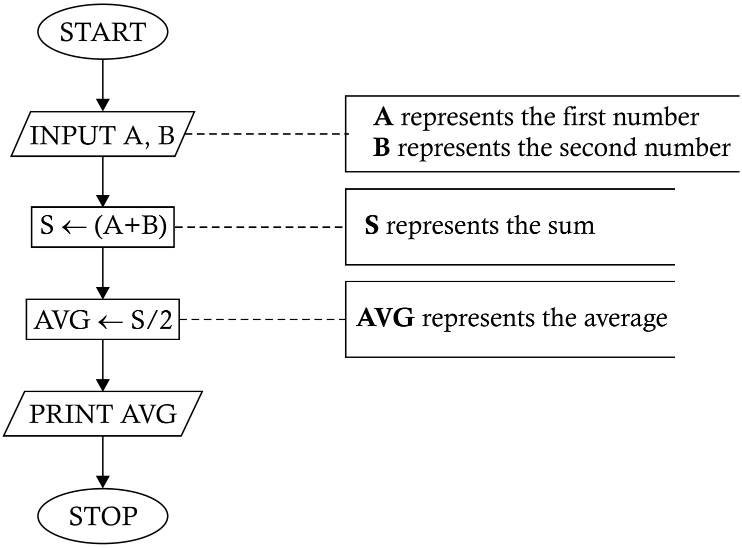



The following algorithm shows the desired procedure:

**Step 1.** INPUT TO A, B
**Step 2.** S ← A+B
(Store the sum of the values in A and B in S)
**Step 3.** PRINT S
(Show the sum obtained in Step 2)
**Step 4.** STOP

A sequence structure shows simple input, output, and process operations.

**Problem 1.2.** *Construct a flowchart to show the procedure for obtaining the average of two given numbers.*



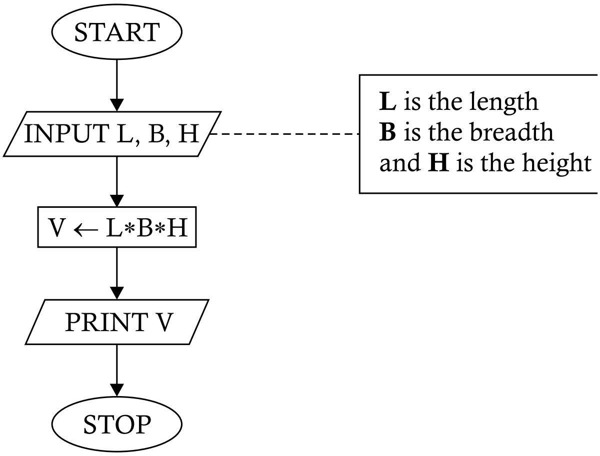

**Task Analysis.** From the concept of determining the average of two given numbers, we know that the given numbers must be added together to obtain the sum first; the sum is then divided by 2 to obtain the average. The flowchart for Problem 1.2 illustrates this idea.

The algorithm corresponding to Problem 1.2 is shown below:

**Step 1.** INPUT TO A, B
**Step 2.** S ← A + B
(Store the sum of the values in A and B and store in S)
**Step 3.** AVG ← S/2
(Compute the average)
**Step 4.** PRINT AVG (Show the average)
**Step 5.** STOP

**Problem 1.3.** *Construct a flowchart to show how to obtain the volume of a rectangular box.*

**Task Analysis.** We know that the formula to determine the volume of a rectangular box is Volume = Length × Breadth × Height. To determine the volume of a rectangular box, we need to know the length, breadth, and height of the box. When these values are multiplied together, the product represents the desired volume.

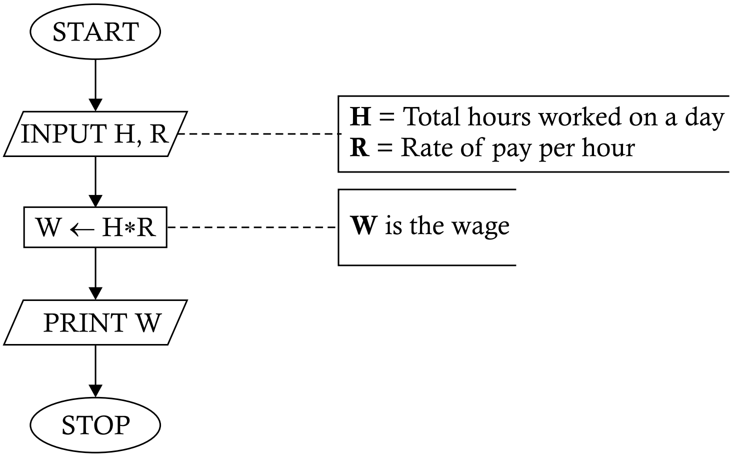



The algorithm for the solution of Problem 1.3 is given below:

**Step 1.** INPUT TO L, B, H
**Step 2.** COMPUTE V ← L*B*H
**Step 3.** PRINT V
**Step 4.** STOP

**Problem 1.4.** *Construct a flowchart to show how to obtain the daily wage of a worker on the basis of the hours worked during the day.*

**Task Analysis.** The daily wage depends on two factors: the hours worked and hourly rate of pay. When the hours worked is multiplied by the rate of pay, the product represents the wage of the worker.



The algorithm for the solution of Problem 1.4. is given below:

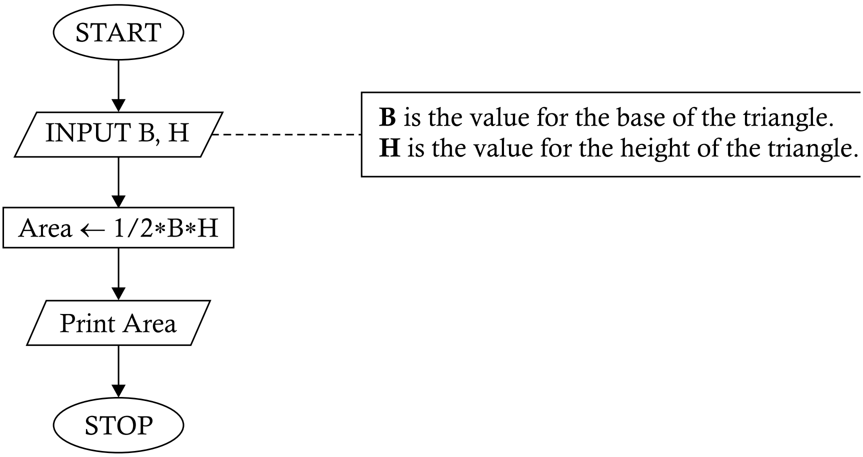
**Step 1.** INPUT TO H, R
**Step 2.** COMPUTE W ← H∗R
(Store the product of the values in H and R in W)
**Step 3.** PRINT W
**Step 4.** STOP

**Problem 1.5.** *Construct a flowchart to show how to obtain the area of a triangle on the basis of the base and height.*

**Task Analysis.** We know that the formula to find out the area of a triangle is

$$\text{Area} = \frac{1}{2} \times \text{base} \times \text{height}$$

The inputs required to obtain the area of a triangle are its base and height. We can then put the values in the above formula to obtain the area.

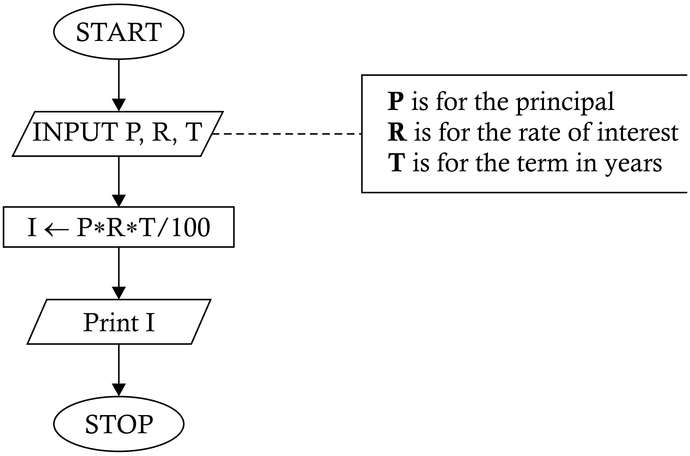



The algorithm corresponding to the above procedure is given below:

**Step 1.** INPUT TO B, H
(B is for the base and H is for the height of the triangle)
**Step 2.** COMPUTE AREA $\leftarrow \frac{1}{2} * B * H$
**Step 3.** PRINT AREA
**Step 4.** STOP

**Problem 1.6.** *Develop a flowchart to show the steps in finding the simple interest on a given amount at a given rate of interest.*

**Task Analysis.** We know that if P is the principal, R is the rate of interest, and T is the term in years, then the simple interest I is given by the formula $I = \frac{P*R*T}{100}$. To determine the simple interest on a given amount, we need the principal amount (P), the rate of interest (R), and the term in years (T). By putting the values in the formula above, we get the desired simple interest.

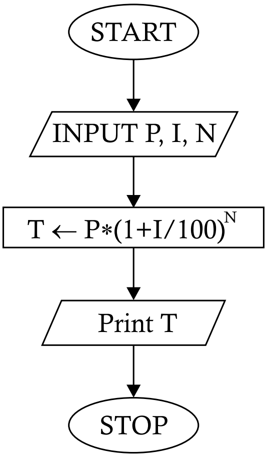



The algorithm corresponding to the above logic is given below:

**Step 1.** INPUT TO P, R, T
**Step 2.** COMPUTE I ← P*R*T / 100
**Step 3.** PRINT I
**Step 4.** STOP

**Problem 1.7.** *If P amount of money is invested for N years at an annual rate of interest I, the money grows to an amount T, where T is given by* $T = P\,(1 + I/100)^N$. *Draw a flowchart to show how T is determined.*

**Task Analysis.** The solution to this problem is very simple, and it is similar to the preceding one. The inputs required are the values for P, I, and N. The output T can then be obtained by putting the values in the formula.




The algorithm corresponding to Problem 1.7 is given below:

**Step 1.** INPUT TO P, I, N

**Step 2.** COMPUTE $T \leftarrow P * \left(1 + \frac{I}{100}\right)^{N}$

**Step 3.** PRINT T

**Step 4.** STOP

**Problem 1.8.** *Construct a flowchart to show how a student's registration number and grades in 3 subjects, $m_1$, $m_2$, and $m_3$, are displayed along with the total average grade.*

**Task Analysis.** The data supplied as inputs are the registration number and *grades* obtained in three subjects. The registration number contributes nothing to the process of deriving the desired output; it just identifies the person about whom the total grade and the average grade are obtained. The total grade can be obtained by taking the sum of the marks $m_1$, $m_2$, and $m_3$, and the average can be obtained by dividing the total by 3. The steps are illustrated below.

The algorithm corresponding to the above problem is given below:

**Step 1.** INPUT TO REGN-NO

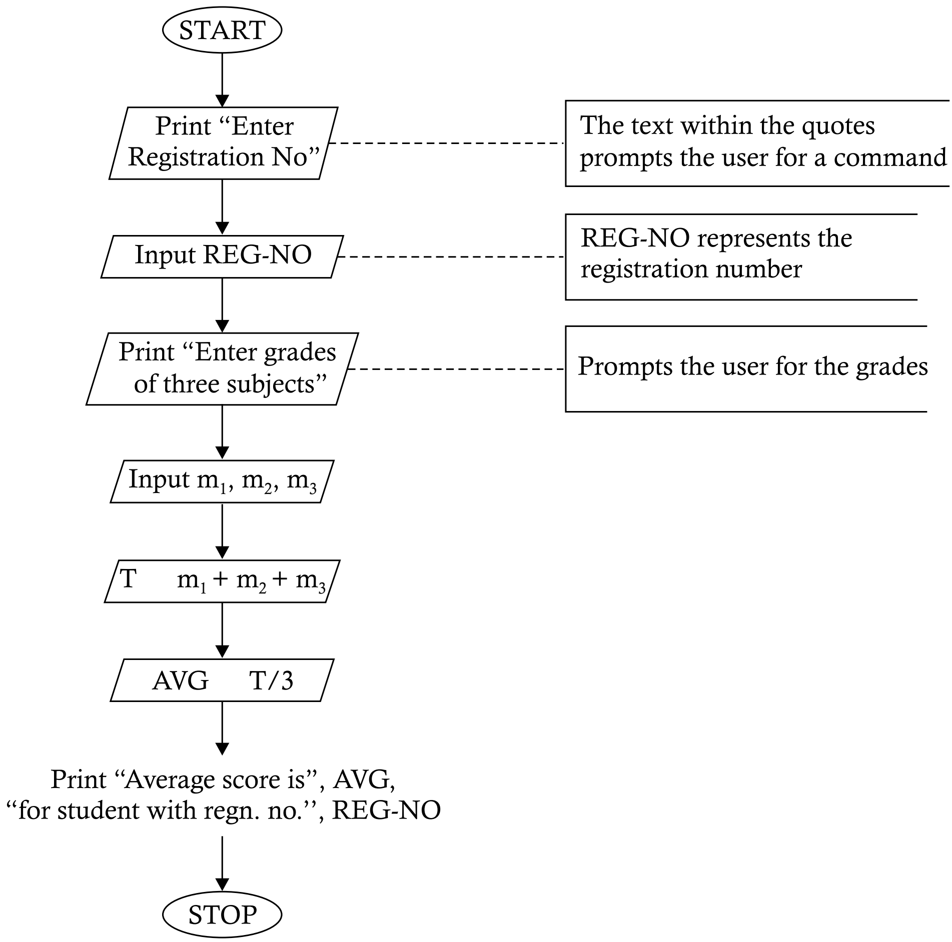
**Step 2.** INPUT TO M1, M2, M3
(M1, M2, and M3 are for holding the grades in three subjects)

**Step 3.** COMPUTE T ← M1 + M2 + M3

**Step 4.** COMPUTE AVG ← T/3

**Step 5.** PRINT REGN-NO, AVG

**Step 6.** STOP.

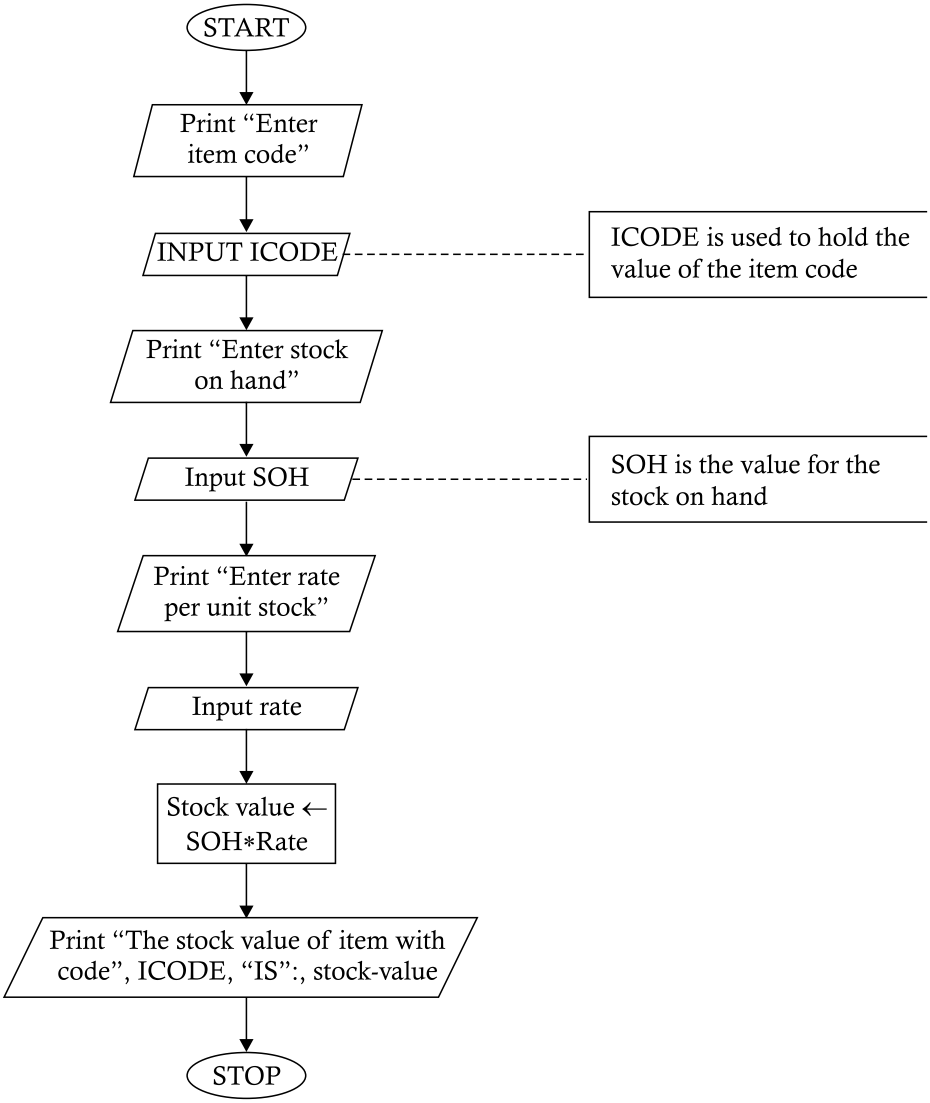



**Problem 1.9.** *Draw a flowchart to accept the item's code, stock on hand, and the rate per unit of stock in a department store and display the stock value of the store.*

**Task Analysis.** The inputs required to determine the stock value of the store are the stock on hand and the rate per unit of stock, which are multiplied together to determine the stock value. The item's code is used as the identification data.

The algorithm corresponding to the solution for Problem 1.9 is as follows:

**Step 1.** INPUT TO I CODE
**Step 2.** INPUT TO SOH (SOH stands for "stock on hand")
**Step 3.** INPUT TO RATE




**Step 4.** COMPUTE STOCK-VALUE ← SOH∗RATE
**Step 5.** PRINT STOCK-VALUE, ICODE
**Step 6.** STOP

**Problem 1.10.** *Draw a flowchart to determine the volume $V_2$ of a certain mass of gas at a pressure $P_2$ if the initial volume is $V_1$ at a pressure $P_1$, keeping the temperature constant.*

**Task Analysis.** From Boyle's law, we know that if the temperature remains constant, the volume of a given mass of gas varies inversely with its pressure. If V is a volume of a given mass of gas at a pressure P, then

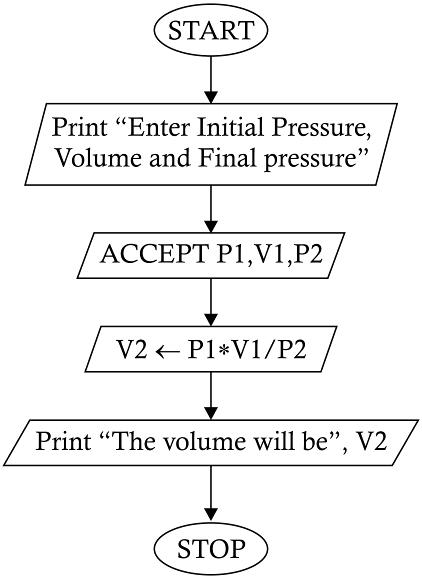
$$V \propto \frac{1}{P}, \text{ at a constant temperature}$$

*i.e.,* PV = constant

Hence, we can write $P_1V_1 = P_2V_2$.

If the initial pressure and volume are known and the final pressure is also known for a given mass of gas, then the final volume $V_2$ can be determined from the formula.

$$V_2 = \frac{P_1V_1}{P_2}, \text{ the temperature being constant.}$$



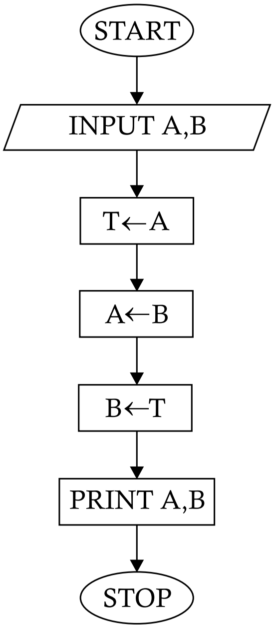
Hence, the inputs are $P_1$, $V_1$, and $P_2$, which gives us $V_2$ with the above formula.

The algorithm corresponding to Problem 1.10 is given below:

**Step 1.** INPUT TO P1, V1, P2
(P1 holds the value for the initial pressure, V1 holds the value of the initial volume, and P2 holds the value of the final pressure.)
**Step 2.** COMPUTE V2 ← P1∗V1/P2
**Step 3.** PRINT V2
**Step 4.** STOP

**Problem 1.11.** *Draw a flowchart to show how to interchange the values of two variables.*

**Task Analysis.** The task of interchanging the values of two variables implies that the values contained by the variables are to be exchanged *i.e.*, the data value contained by the first variable should be contained by the second variable and that by the second variable should be contained by the first variable. If A and B are two variables, and if the values contained by them are 10 and 20 respectively, the problem is to make the contents of A and B, 20 and 10, respectively. This can be done simply with the help of a third variable used as an intermediate variable. The third variable holds the value of either A or B,



so that if the value of one variable is assigned to the other, the assignee's value is not lost forever but is available in the intermediate variable. Hence, it can then be assigned to the other variable.

The algorithm of the problem is as follows:

**Step 1.** ACCEPT A,B
**Step 2.** T ← A (Assign value in A to T)
**Step 3.** A ← B (Assign value in B to A)
**Step 4.** B ← T (Assign value in T to B)
**Step 5.** PRINT A,B
**Step 6.** STOP

## EXERCISES

**Construct flowcharts to show the steps involved to accomplish the following:**

**(i)** Find the product of two numbers.

**(ii)** Find the remainder when one number is divided by the other.

**(iii)** Find the area of a parallelogram.

**(iv)** Find the area of the four walls of a rectangular room.

**(v)** Find the area and perimeter of a circular plot.

**(vi)** Find the area of a triangle based on the length of three sides.

**(vii)** Find the area and perimeter of a square.

**(viii)** Find the cost of fencing a rectangle at a given rate.

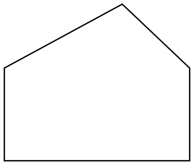
**(ix)** Find the surface area of a cone.

**(x)** Find the volume and surface area of a sphere.

**(xi)** Convert meters to kilometers.

**(xii)** Accept the rate for a dozen bananas and the quantity required to determine the cost.

**(xiii)** Find the cost of a flat having the floor space of the following pattern:



**(xiv)** Determine the acceleration due to gravity ($g$), where g can be obtained from the following formula:

$$T = 2\pi\sqrt{\frac{l}{g}}$$

where T = Time period of a simple pendulum

and $I$ = Effective length of the simple pendulum

**(xv)** Obtain the equivalent Fahrenheit temperature of a temperature given in Celsius where the relationship between the two scales of temperature is

$$\frac{C}{5} = \frac{F-32}{9},$$

where C = Temperature in Celsius

F = Temperature in Fahrenheit