Algorithm and Flowchart

Logic programming

- **Logic programming**

  is a computer programming paradigm in which program statements express facts and rules about problems within a system of formal logic. Rules are written as logical clauses with a head and a body; for instance, "H is true if B1, B2, and B3 are true." Facts are expressed similar to rules, but without a body; for instance, "H is true."

- A typical programming task can be divided into two phases:

- **Problem solving phase-**
  - produce an ordered sequence of steps that describe solution of problem
  - this sequence of steps is called an algorithm

- **Implementation phase-**
  - implement the program in some programming language

Algorithm

- The term algorithm originally referred to any computation performed via a set of rules applied to numbers written in decimal form.

- The word is derived from the phonetic pronunciation of the last name of Abu Ja'far Mohammed ibn Musa al-Khowarizmi, who was an Arabic mathematician who invented a set of rules for performing the four basic arithmetic operations (addition, subtraction, multiplication and division) on decimal numbers.

- **Definition**: *A procedure of a finite sequence of well-defined instructions, each of which can be mechanically carried out in a finite amount of time.*

- **Definition**: *An algorithm is procedure consisting of a finite set of unambiguous rules (instructions) which specify a finite sequence of operations that provides the solution to a problem, or to a specific class of problems for any allowable set of input quantities (if there are inputs). In other word, an algorithm is a step-by-step procedure to solve a given problem*

Algorithm

Algorithm has the following characteristics:

- **Input**: An algorithm may or may not require input
- **Output**: Each algorithm is expected to produce at least one result
- **Definiteness**: Each instruction must be clear and unambiguous.
- **Finiteness**: If the instructions of an algorithm are executed, the algorithm should terminate after finite number of steps
- **Independent** to any programming language

- The algorithm and flowchart include following three types of control structures.

1. Sequence: In the sequence structure, statements are placed one after the other and the execution takes place starting from up to down.
2. Branching (Selection): In branch control, there is a condition and according to a condition, a decision of either TRUE or FALSE is achieved.
3. Loop (Repetition): The Loop or Repetition allows a statement(s) to be executed repeatedly based on certain loop condition e.g. WHILE, FOR loops.

HOW TO WRITE ALGORITHMS

Step 1: Define your algorithms input: Many algorithms take in data to be processed, e.g. to calculate the area of rectangle input may be the rectangle height and rectangle width.

Step 2: Define the variables: Algorithm's variables allow you to use it for more than one place. We can define two variables for rectangle height and rectangle width as HEIGHT and WIDTH (or H & W). We should use meaningful variable name e.g. instead of using H & W use HEIGHT and WIDTH as variable name.

Step-3: Outline the algorithm's operations: Use input variable for computation purpose, e.g. to find area of rectangle multiply the HEIGHT and WIDTH variable and store the value in new variable (say) AREA. An algorithm's operations can take the form of multiple steps and even branch, depending on the value of the input variables.

Step-4: Output the results of your algorithm's operations: In case of area of rectangle output will be the value stored in variable AREA. if the input variables described a rectangle with a HEIGHT of 2 and a WIDTH of 3, the algorithm would output the value of 6.

STEPS IN PROBLEM SOLVING

- First produce a general algorithm (one can use pseudo code)

- Refine the algorithm successively to get step by step detailed algorithm that is very close to a computer language.

- Pseudo code is an artificial and informal language that helps programmers develop algorithms. Pseudo code is very similar to everyday English

Example 1:

- Write an algorithm to determine a student's final grade and indicate whether it is passing or failing. The final grade is calculated as t

Pseudo code:

- Input a set of 4 marks
- Calculate their average by summing and dividing by 4
- if average is below 50

  Print "FAIL"
- else

  Print "PASS"

Detailed Algorithm

- Step 1: Input M1,M2,M3,M4
- Step 2: GRADE (M1+M2+M3+M4)/4
- Step 3: if (GRADE < 50) then

  Print “FAIL”

  else

  Print “PASS” endif

Advantages of algorithm

- It is a step-wise representation of a solution to a given problem, which makes it easy to understand.
- An algorithm uses a definite procedure.
- It is not dependent on any programming language, so it is easy to understand for anyone even without programming knowledge.
- Every step in an algorithm has its own logical sequence so it is easy to debug

Example-2

Design an algorithm and the corresponding flowchart for adding the test scores as given below:

26, 49, 98, 87, 62, 75

Algorithm

- 1. Start
- 2. Sum = 0
- 3. Get the first testscore
- 4. Add first testscore to sum
- 5. Get the second testscore
- 6. Add to sum
- 7. Get the third testscore
- 8. Add to sum
- 9. Get the Forth testscore
- 10. Add to sum
- 11. Get the fifth testscore
- 12. Add to sum
- 13. Get the sixth testscore
- 14. Add to sum
- 15. Output the sum 16. Stop

Example-3

- **Write an algorithm to find the largest among three different numbers entered by the user**
- Step 1: Start
- Step 2: Declare variables a,b and c.
- Step 3: Read variables a,b and c.
- Step 4: If a > b

If a > c

Display a is the largest number.

Else

Display c is the largest number.

Else

If b > c

Display b is the largest number.

Else

Display c is the greatest number. Step 5: Stop

Problems for Practices and file

1. **Write an algorithm to find all roots of a quadratic equation $ax^2+bx+c=0$.**
2. **Write an algorithm to find the factorial of a number entered by the user.**
3. **Write an algorithm to check whether a number entered by the user is prime or not.**
- **4. Write an algorithm to find the Fibonacci series till term≤1000.**