Introduction to Algorithms and Flowcharts

Course Title: Computer Programming I

Course Code: CSE-1122

Saifur Rahaman
Assistant Professor
Dept. of CSE, IIUC

Algorithm: An algorithm is a set of well-defined instructions that designed to solve a specific task.

Rules for constructing an Algorithm[1]

When you are going to create an algorithms, keep following point in mind as:

- **Input**: There should be zero or more values which are to be supplied.
- **Output**: At least one output is to be produced.
- **Definiteness:** Each step must be clear and unambiguous.
- **Finiteness:** If we trace the steps of an algorithm, then for all cases, the algorithm must terminate after a finite number of steps.
- **Effectiveness:** Each step must be sufficiently basic that a person using only paper and pencil can in principle carry it out. In addition, not only each step is definite, it must also be feasible.
- **Comment Session:** Comment session is not mandatory for writing an algorithm. Comment is additional info of program for easily modification. In algorithm comment would be appear between two square brackets **[]**. *For example:* [this is a comment of an algorithm]

Let's understand to algorithm by example:

Q_1. Write algorithm to calculate the sum and average of two numbers.

Ans.:

[Procedure for calculate sum and average of two numbers]
Step1 : Start
Step2 : Read two numbers n,m
Step3 : Calculate sum=n+m
Step4 : Calculate avg=sum/2
Step5 : Print sum, avg
Step5 : Stop
[End of procedure for calculate sum and average of two numbers]

Q_2. Write an algorithm to convert a decimal number into binary.

Ans.:

[procedure for convert decimal to binary number]
Step1 : Start
Step2 : Read number num
Step3 : Set x=1
Step4 : B(x)= num MOD 2
Step5 : num=num/2
Step6 : If num is equal to 0 then goto step8
Step7 : x=x+1
Step8 : goto step3
Step9 : Print B(x)
Step10 : x=x-1
Step11 : If x is greater than 0 then goto step8
step12 : Stop
[end of procedure for convert decimal to binary number]

Q_3. Write a C program to accept a number from user and print the sum of square. Also write down the algorithm and draw flowchart.

Ans.:

Algorithm for accept a number from user and calculate sum of square:

[Sum of square procedure: accept number num from user, and set sum=0 and calculate sum of square.]
Step 1. Start
Step 2. Read number num
Step 3. [Initialize]
sum=0, i=1
Step 4. Repeat step 4 through 6 until i<=num
Step 5. sum=sum+(i*i)
Step 6. i=i+1
Step 7. print the sum of square
Step 8. stop
[end of loop step 4]
[end of sum of square procedure]

Flowchart:

A **flowchart** is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.[2]

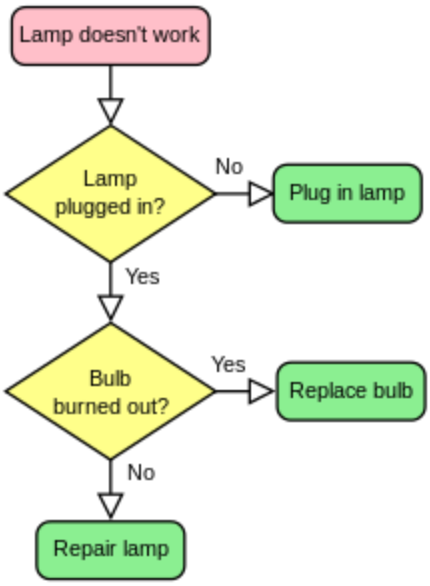
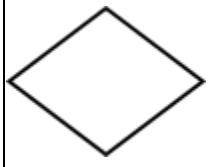
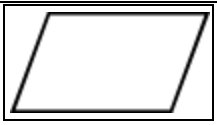
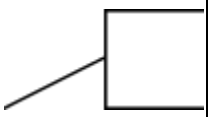
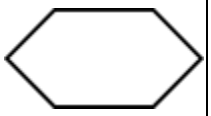
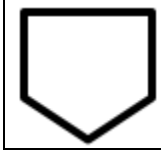


Figure 1: A simple flowchart representing a process for dealing with a non-functioning lamp.

The following are some of the commonly used shapes used in flowcharts. Generally, flowcharts flow from top to bottom and left to right.

Shape	Name	Description
	Flow Line	An arrow coming from one symbol and ending at another symbol represents that control passes to the symbol the arrow points to. The line for the arrow can be solid or dashed. The meaning of the arrow with dashed line may differ from one flowchart to another and can be defined in the legend.
	Terminal	Represented as circles, ovals, stadiums or rounded (fillet) rectangles. They usually contain the word "Start" or "End", or another phrase signaling the start or end of a process, such as "submit inquiry" or "receive product".
	Process	Represented as rectangles. This shape is used to show that something is performed. Examples: "Add 1 to X", "replace identified part", "save changes", etc....
	Decision	Represented as a diamond (rhombus) showing where a decision is necessary, commonly a Yes/No question or True/False test. The conditional symbol is peculiar in that it has two arrows coming out of it, usually from the bottom point and right point, one corresponding to Yes or True, and one corresponding to No or False. (The arrows should always be labeled.) More than two arrows can be used, but this is normally a clear indicator that a complex decision is being taken, in which case it may need to be broken-down further or replaced with the "predefined process" symbol. Decision can also help in the filtering of data.

	Input/Output	Represented as a parallelogram. Involves receiving data and displaying processed data. Can only move from input to output and not vice versa. Examples: Get X from the user; display X.
	Annotation	Annotations represent comments or remarks about the flowchart. Like comments found in high-level programming languages, they have no effect on the interpretation or behavior of the flowchart. Sometimes, the shape consists of a box with dashed (or dotted) lines.
	Predefined Process	Represented as rectangles with double-struck vertical edges; these are used to show complex processing steps which may be detailed in a separate flowchart. Example: PROCESS-FILES. One subroutine may have multiple distinct entry points or exit flows (see coroutine). If so, these are shown as labeled 'wells' in the rectangle, and control arrows connect to these 'wells'.
	Preparation	Represented as a hexagon. May also be called initialization. Shows operations which have no effect other than preparing a value for a subsequent conditional or decision step. Alternatively, this shape is used to replace the Decision Shape in the case of conditional looping.
	On-Page Connector	Generally represented with a circle, showing where multiple control flows converge in a single exit flow. It will have more than one arrow coming into it, but only one going out. In simple cases, one may simply have an arrow point to another arrow instead. These are useful to represent an iterative process (what in Computer Science is called a loop). A loop may, for example, consist of a connector where control first enters, processing steps, a conditional with one arrow exiting the loop, and one going back to the connector. For additional clarity, wherever two lines accidentally cross in the drawing, one of them may be drawn with a small semicircle over the other, showing that no connection is intended.
	Off-Page Connector	Represented as a home plate-shaped pentagon. Similar to the on-page connector except allows for placing a connector that connects to another page.

Flowchart for sum of square C program as following:

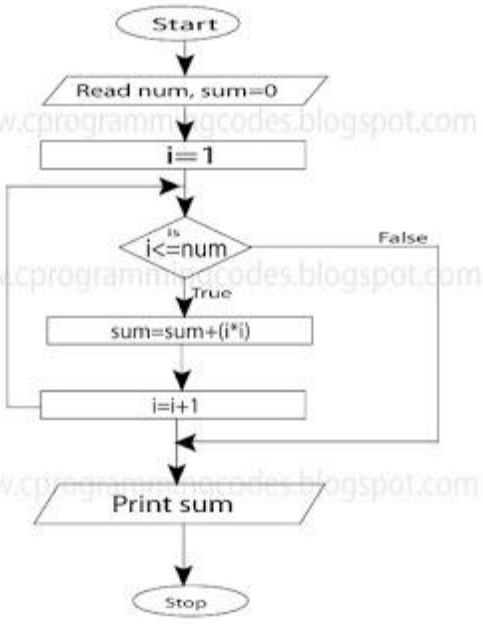


Figure: Flowchart for calculate sum of square C program

```
/*c program for accept number and calculate sum of square*/
#include<stdio.h>
int main()
{
 int num,sum=0,i;
 printf("Enter any number : ");
 scanf("%d", &num);
 for(i=1; i<=num; i++)
    sum = sum + (i*i);
 printf("Sum of square of %d = %d",num,sum);
 return 0;
}
```

Q_4. Write a C program to find the factorial value of a number. Also write the algorithm and draw flowchart.

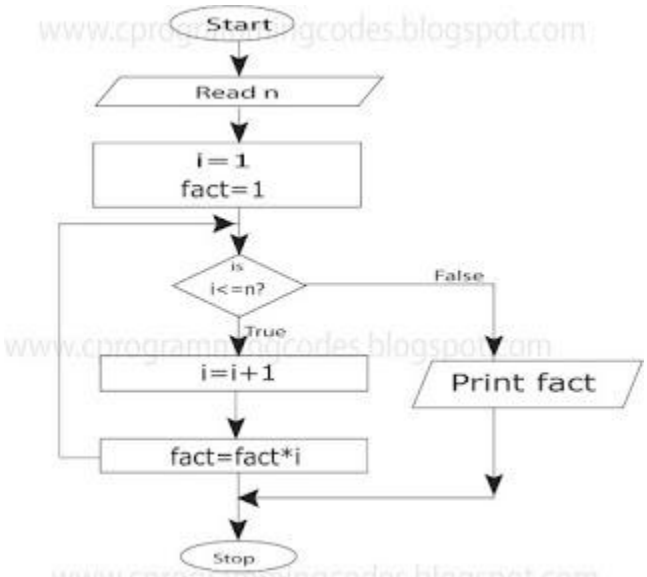
Ans.:

Algorithm for calculate factorial value of a number:

[algorithm to calculate the factorial of a number]
step 1. Start
step 2. Read the number n
step 3. [Initialize]
        i=1, fact=1
step 4. Repeat step 4 through 6 until i=n

step 5. fact=fact*i
step 6. i=i+1
step 7. Print fact
step 8. Stop
[process finish of calculate the factorial value of a number]

Flowchart for calculate factorial value of a number:




```
/*c program to find out factorial value of a number*/
#include<stdio.h>
#include<conio.h>
int main()
{
 int n,i,fact=1;
 printf("Enter any number : ");
 scanf("%d", &n);
 for(i=1; i<=n; i++)
    fact = fact * i;
 printf("Factorial value of %d = %d", n, fact);
 return 0;
}
```

References:

1. http://cprogrammingcodes.blogspot.com/2011/11/algorithm.html
2. https://en.wikipedia.org/wiki/Flowchart#cite_note-SSEV-1