Program Development Process

J. Kamiri

Algorithm

- This is step by step procedure of solving a problem, attaining a goal or accomplishing a certain objective or verifying a problem has been solved.
- A step-by-step (a set of) instructions which when followed will produce a solution to a given problem.

Characteristics of a good algorithm

- **Definiteness-**Each step of an algorithm must be clearly defined or the action must clearly specified.
- **Finiteness-**An algorithm must terminate (end) after a number of steps.
- **Efficiency-**Algorithms must be effective, or operations are executable, avoid ambiguity.
- **Input**
- **Output**

Program design Tools.

Algorithms can be illustrated using the following tools:

- Pseudocodes.
- Flowcharts.
- Decision Tables.
- Decision Trees.

Note. For any given problem, the programmer must choose which algorithm (method) is best suited to solve it.

Pseudocode-

This is an artificial and informal language that helps programmers develop algorithms. It is similar to every day English.

- A *pseudocode* is a method of documenting a program logic in which English-like statements are used to describe the processing steps.
- These are structured English-like phrases that indicate the program steps to be followed to solve a given problem.

It is convenient and user friendly although it’s not actual computer programming language. This helps programmers in “thinking out” a program before putting it in actual coding.

Guidelines for designing a good pseudocode.

- The statements must be short, clear and readable.
- The statements must not have more than one meaning (i.e., should not be ambiguous).
- The pseudocode lines should be clearly outlined and indented.
- A pseudocode must have a **Begin** and an **end**.

i.e., a pseudocode should show clearly the start and stop of executable statements and the control structures.

The input, output and processing statements should be clearly stated using keywords such as PRINT, READ, and INPUT

Example 1

- *Write a pseudocode that can be used to prompt the user to enter two numbers, calculate the sum and average of the two numbers and then display the output on the screen.*

```
START
    PRINT “Enter two numbers”
    INPUT X, Y
    Sum = X + Y
    Average = Sum/2
    PRINT Sum
    PRINT Average
STOP
```

Example 2

Write a structured algorithm that would prompt the user to enter the Length and Width of a rectangle, calculate the Area and Perimeter, then display the result.

Solution

Step 1: Draw the rectangle of Length (L) and Width (W).

Step 2: Write down the Pseudocode.

```
START
    PRINT "Enter Length and Width"
    READ L, W
    Area = L * W
    Perimeter = 2 (L + W)
    PRINT Area
    PRINT Perimeter
STOP
```

Example 3

Solve the following:

Write a pseudocode that can be used to input the Diameter, then calculate Circumference and Area of a circle and then display the output on the screen.

- It is important to use program control structures when writing Pseudocodes. The most common constructs are:
  - **(i). Looping (Repetition / Iteration) –** used where instructions are to be repeated under certain conditions.
  - **(ii). Selection –** used when choosing a specified group of instructions for execution. The group chosen depends on certain conditions being satisfied.

Example 5: Using control structures

Write a pseudocode for a program that can be used to classify people according to age. If a person is more than 20 years; output "Adult" else output "Young person".

```
START
    PRINT "Enter the Age"
    INPUT Age
    IF Age > 20 THEN
    PRINT "Adult"
    ELSE
    PRINT "Young person"
    End IF
STOP
```

FLOWCHART

Flowchart

- This is a graphical representation of an algorithm.
- They are drawn using special symbols. They are useful for developing and representing an algorithm. They clearly show how control structures operates.
- A **Flowchart** is a diagrammatic or pictorial representation of a program's algorithm.
- It is a chart that demonstrates the logical sequence of events that must be performed to solve a problem

Types of Flowcharts

System flowchart.

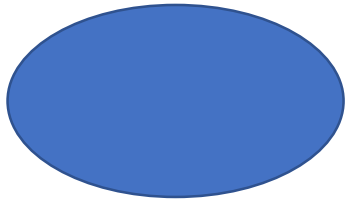
- A *System flowchart* is a graphical model that illustrates each basic step of a data processing system.
- It illustrates (in summary) the sequence of events in a system, showing the department or function responsible for each event.

Program flowchart.

- This is a diagram that describes, in sequence, all the operations required to process data in a computer program.
- A *program flowchart* graphically represents the types of instructions contained in a computer program as well as their sequence & logic.

Flowchart symbols and Notations

Terminal Symbol:

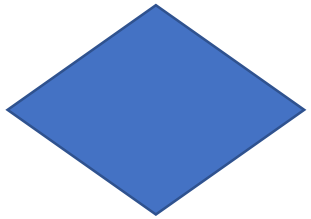


Start/ End

Process

Input/output

Connector



Decision

Flowchart

Start/End:

- It is used to indicate the point at which a flowchart, a process or an algorithm begins & ends.
- All Flowcharts must have a START & STOP symbol.
- The words **Begin** & **End** (or **Start** & **Stop**) should be inserted in the Terminal symbol.

Input or Output symbol :

- It is used to identify/specify an input operation or output operation.
- The words mostly associated with I/O operations are **READ** & **PRINT**. READ describes the entry of computer data, while PRINT relates to the printed output of information.

Cont.

Example of Input and output

Process Symbol:

- Used to indicate process or data transformation is taking place
- The information placed within the process symbol may be an algebraic formula or a sentence to describe processing.
- Example: SUM=A+B

Cont.

Decision Symbol

- It is used to indicate/ specify a condition or to show the decision to be made.
- There are 2 main components of a Decision symbol:

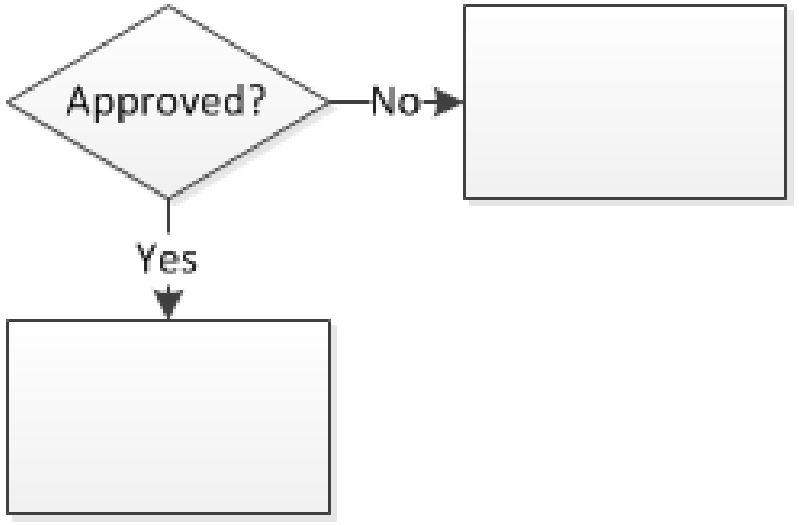
  (i). A question asked within the Decision symbol, that indicates the comparison / logical operation.

  (ii). The results of the comparison (which are given in terms of **YES** or **NO**).

- The arrows labeled YES or NO lead to the required action corresponding to the answer to the question.

Cont.

The arrows labeled YES or NO lead to the required action corresponding to the answer to the question.



Cont.

Flow lines:

- Flow lines with arrowheads are used to indicate the direction of processing of the program logic,
- i.e., they show the order in which the instructions are to be executed.
- The normal flow of a flowchart is from *Top* to *Bottom*, and *Left* to *Right*.
- Flow lines should never cross each other.

Cont.

Connector:

- Sometimes, a flowchart becomes too long to fit in a single page, such that the flow lines start crisscrossing at many places causing confusion & also making the flowchart difficult to understand.
- The **Connector symbol** is used as a connecting point for arrows coming from different directions.
- A Connector symbol is represented by a Circle, and a letter or digit is placed within the circle to indicate the link.
- Connectors do not represent any operation. They are used to connect two parts of a flowchart, indicating that the flow of data is not broken.

General guidelines for drawing a program flowchart.

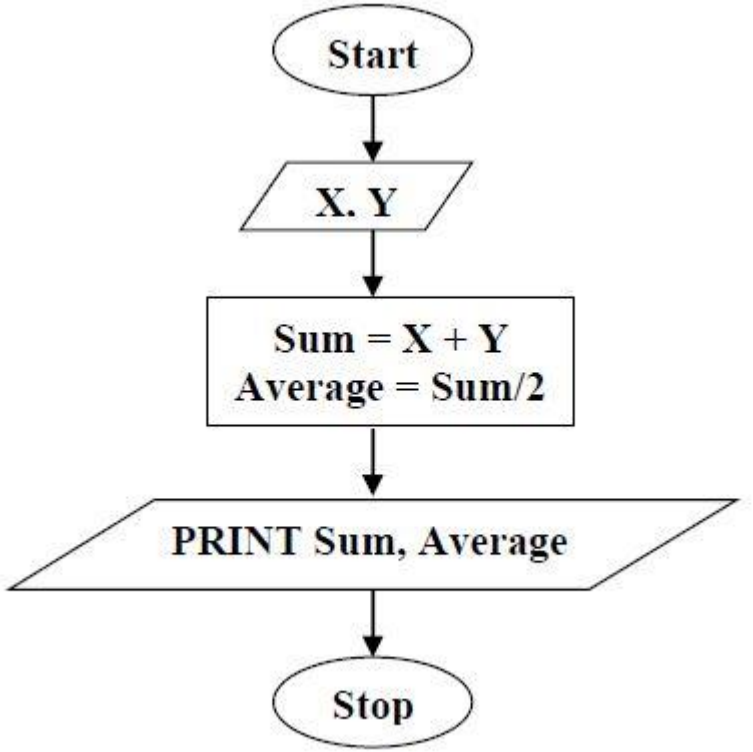
- A flowchart should have only one entry/starting point and one exit point (i.e., ensure that the flowchart has a logical start and finish).
- The flowchart should be clear, neat and easy to follow.
- Use the correct symbol at each stage in the flowchart.
- The flowchart should not be open to more than one interpretation.
- Avoid overlapping the lines used to show the flow of logic as this can create confusion in the flowchart.

Cont.

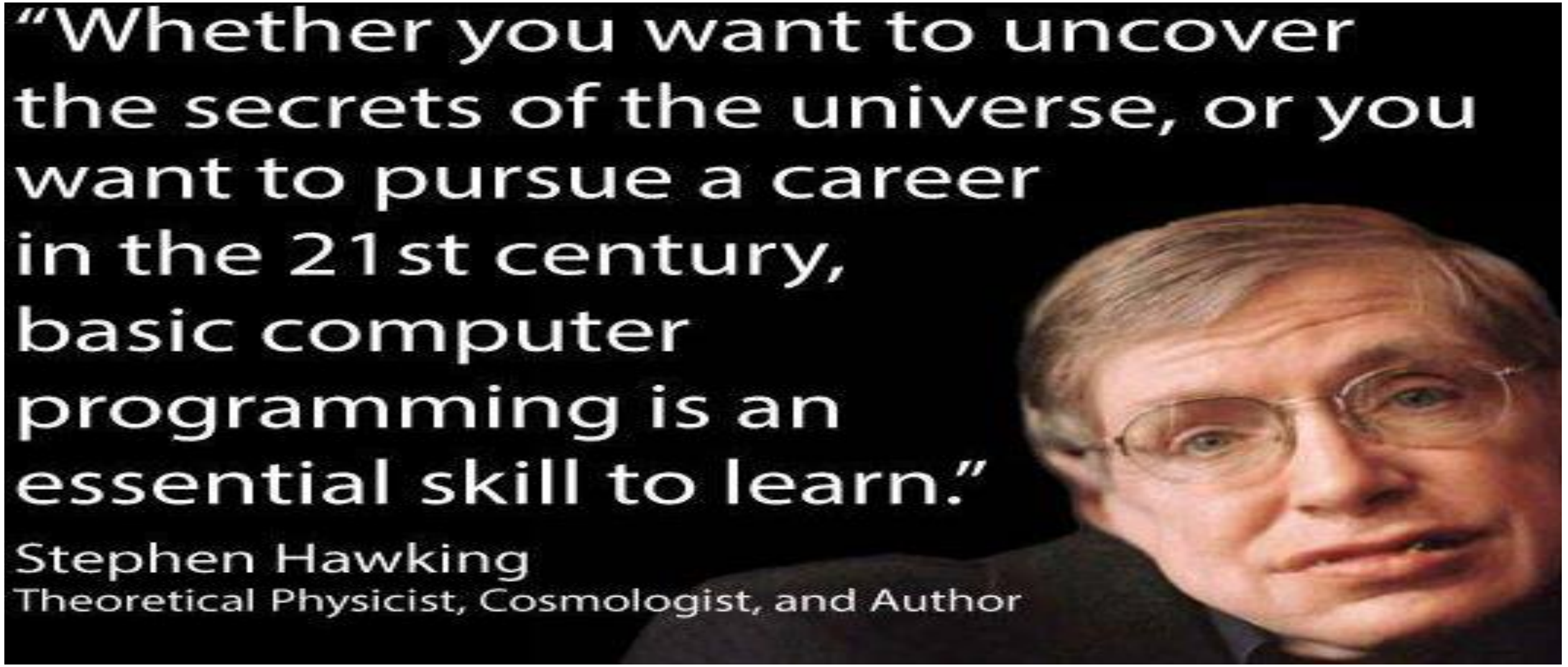
- Make comparison instructions simple, i.e., capable of YES/NO answers.
- The logical flow should be clearly shown using arrows.
  - A flowchart should flow from the Top to Bottom of a page, and from the Left to the Right.
- Where necessary, use Connectors to reduce the number of flow lines.
- Check to ensure that the flowchart is logically correct & complete.

Example

Draw a flowchart for a program that can be used to prompt the user to enter two numbers, find the sum and average of the two numbers and then display the output on the screen.



The End!


"Whether you want to uncover the secrets of the universe, or you want to pursue a career in the 21st century, basic computer programming is an essential skill to learn."
Stephen Hawking
Theoretical Physicist, Cosmologist, and Author