

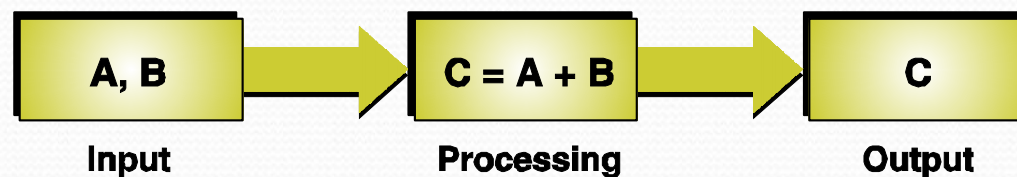
Computer Programming and Languages

Introduction

Developing a Program

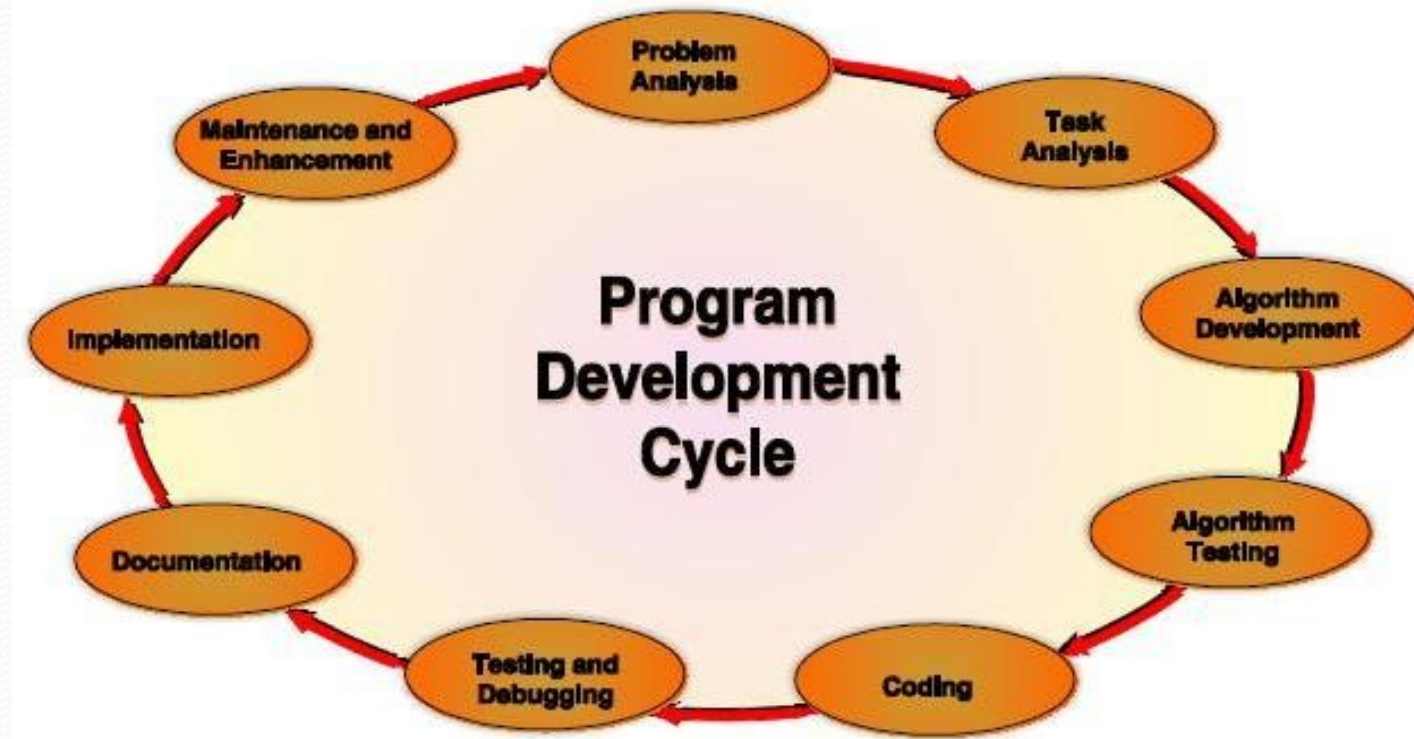
- ✓ Computers work on a set of instructions called computer program, which clearly specifies the way to carry out a task.
- ✓ In order to design a program, a programmer must determine three basic rudiments:
 - The instructions to be performed.
 - The order in which those instructions are to be performed.
 - The data required to perform those instructions.

- ✓ To perform a task using a program, a programmer has to consider various inputs of the program along with the process, which is required to convert the input into desired output.
- ✓ Suppose we want to calculate the sum of two numbers, A and B and store the sum in C. Here, A and B are inputs, addition is process and C is the output of the program.



Introduction (Contd.)

Program Development Cycle



Algorithm

- ✓ An algorithm is a finite sequence of explicit instructions which when provided with a set of input values produces an output and then terminates.
- ✓ It provides a logical structure to plan the solution. Once the solution is properly designed, the only job left is to code that logic into the respective programming language.
- ✓ To be an algorithm, the steps must be unambiguous and after a finite number of steps, the solution of the problem should be achieved.

Example:

Algorithm: To determine the largest of three numbers

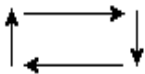

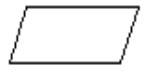




1. Start
2. Read three numbers A, B, C
3. Find the larger number between A and B and store it in MAX_AB
4. Find the larger number between MAX_AB and C and store it in MAX
5. Display MAX
6. Stop

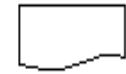
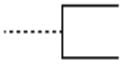
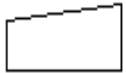
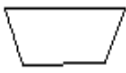




Flowchart

- ✓ A flowchart is a pictorial representation of an algorithm in which the steps are drawn in the form of different shapes of boxes and the logical flow is indicated by interconnecting arrows.
- ✓ The boxes represent operations and the arrows represent the sequence in which the operations are implemented.
- ✓ **Benefits of Flowcharts**
 - Makes logic clear
 - Communication
 - Effective analysis
 - Useful in Coding
 - Proper Testing and Debugging
 - Appropriate Documentation
- ✓ **Limitations of Flowcharts**
 - Complex
 - Costly
 - Difficult to modify
 - No update

Flowchart (Contd.)

Flowchart Symbols

Symbol	Symbol Name	Description
	Flow Lines	Flow lines are used to connect symbols. These lines indicate the sequence of steps and the direction of flow of control.
	Terminal	This symbol is used to represent the beginning (start), the termination (end) or halt (pause) in the program logic.
	Input/Output	It represents information entering or leaving the system, such as customer order (input) and servicing (output).
	Processing	Process symbol is used for representing arithmetic and data movement instructions. It can represent a single step ('add two cups of flour'), or an entire sub-process ('make bread') within a larger process.
	Decision	Decision symbol denotes a decision (or branch) to be made. The program should continue along one of the two routes (IF/ELSE). This symbol has one entry and two exit paths. The path chosen depends on whether the answer to a question is yes or no.
	Connector	Connector symbol is used to join different flow lines.
	Off-page Connector	This symbol is used to indicate that the flowchart continues on the next page.

Symbol	Symbol Name	Description
	Document	Document is used to represent a paper document produced during the flowchart process.
	Annotation	It is used to provide additional information about another flowchart symbol. The content may be in the form of descriptive comments, remarks or explanatory notes.
	Manual Input	Manual input symbol represents input to be given by a developer/programmer.
	Manual Operation	Manual operation symbol shows that the process has to be done by a developer/programmer.
	Online Storage	This symbol represents the online data storage such as hard disks, magnetic drums, or other storage devices.
	Offline Storage	This symbol represents the offline data storage such as sales on OCR and data on punched cards.
	Communication Link	Communication link symbol is used to represent data received or to be transmitted from an external system.
	Magnetic Disk	This symbol is used to represent data input or output from and to a magnetic disk.

Flowchart (Contd.)

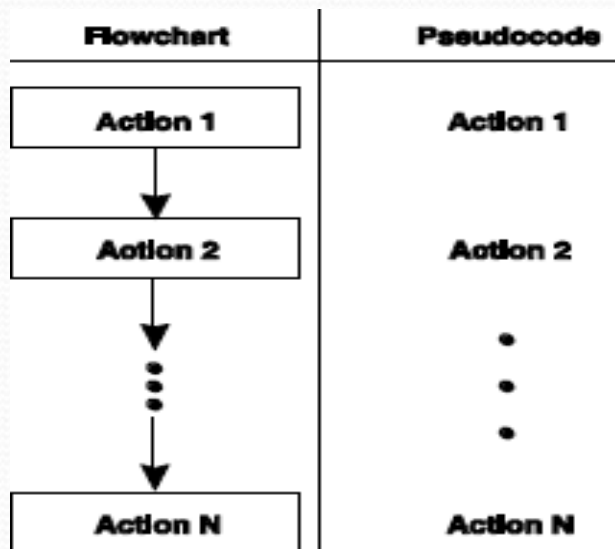
Guidelines for Preparing Flowcharts

- ✓ The flowchart should be clear, neat, and easy to follow.
- ✓ The flowchart must have a logical start and finish.
- ✓ The direction of the flow of a procedure should always be from left to right or top to bottom.
- ✓ Only one flow line should come out from a process symbol.
- ✓ Only one flow line should enter a decision symbol. However, two or three flow lines (one for each possible answer) may leave the decision symbol.
- ✓ Only one flow line is used with a terminal symbol.
- ✓ In case of complex flowcharts, connector symbols are used to reduce the number of flow lines.
- ✓ Intersection of flow lines should be avoided to make it a more effective and better way of representing communication.

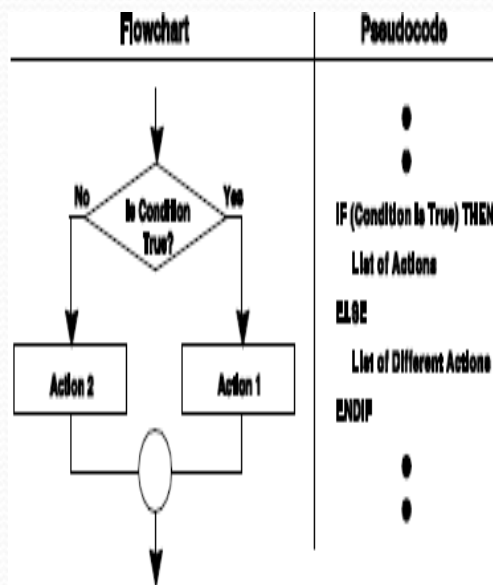
Program Control Structures

- ✓ Control structures are used to express algorithms, flowcharts, decision tables, and pseudocodes as actual computer programs. Essentially, there are three control structures:

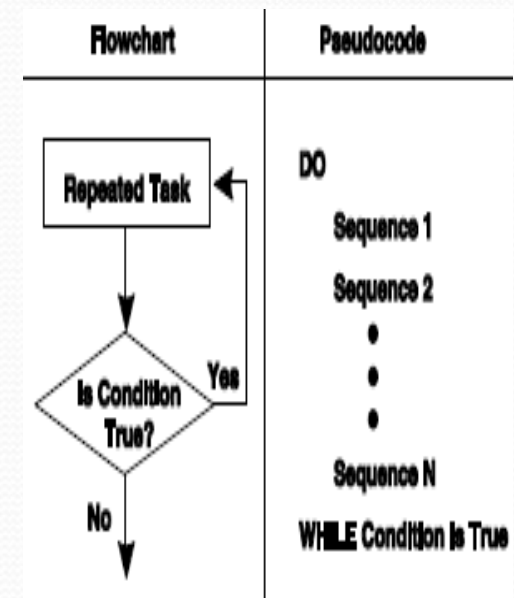
✓ SequenceControl Structure



✓ SelectionControl Structure



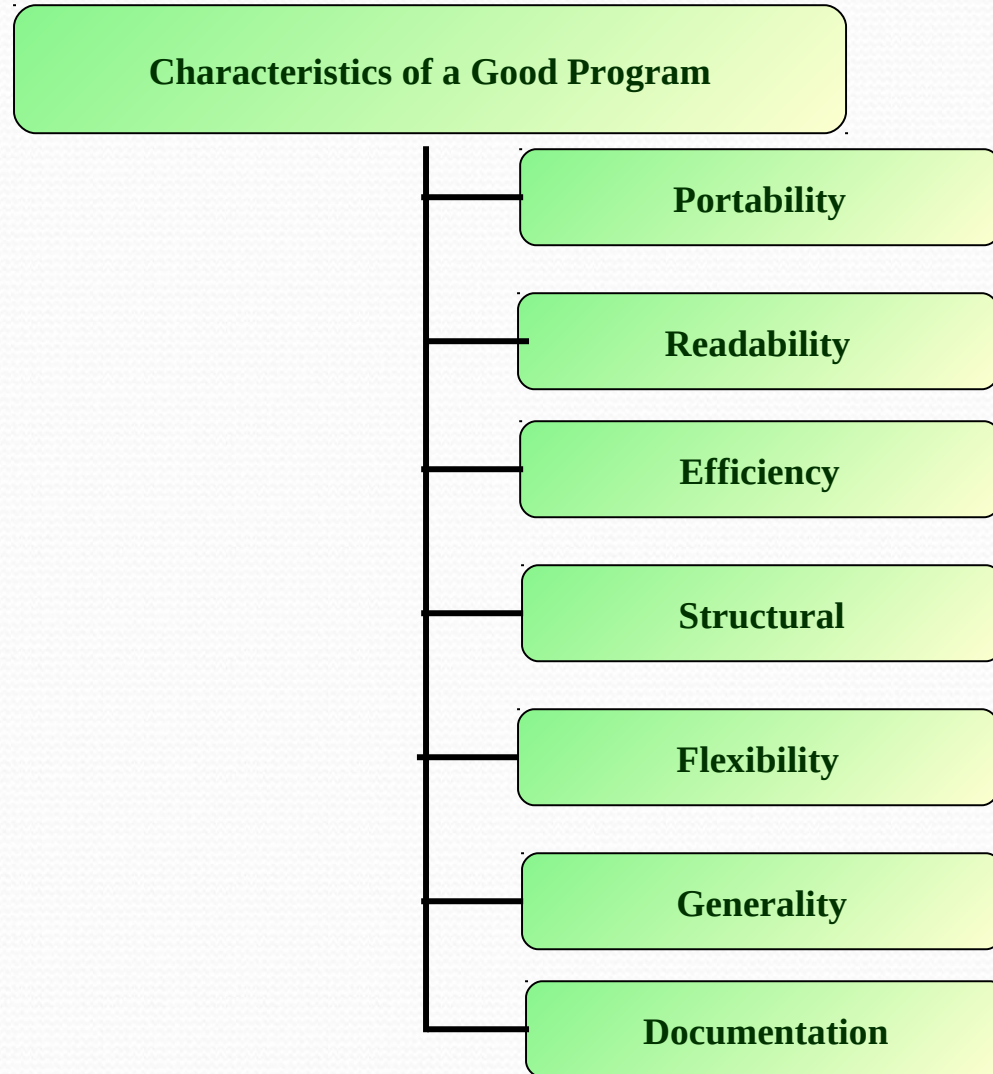
✓ RepetitionControl Structure



Programming Paradigms

- ✓ Programming paradigm refers to the approach used to develop the program for solving a problem. Broadly, programming can be classified in the following three categories:
 - **Unstructured Programming:** This refers to writing small and simple programs consisting of only one main program. All the actions such as providing input, processing, and displaying output are done within one program only.
 - **Structured Programming:** In this programming, a program is broken down into small independent tasks that are small enough to be understood easily. Each task is developed independently and after completion, they are combined together to solve the problem. Structured programming can be performed in either of two ways: procedural or modular.
 - **Object-oriented Programming:** In this programming, programs are organized as cooperative collection of objects, each of which represents an instance of some class, and whose classes are members of a hierarchy of classes united by way of inheritance relationship.
 - Object
 - Abstraction
 - Polymorphism
 - Class
 - Encapsulation
 - Inheritance

Characteristics of a Good Program



Programming Languages

Types of Programming Languages

- ✓ programming language consists of a set of characters, symbols, and usage rules that allow the user to communicate with computers just as natural languages used for communication among human beings.
 - **Machine Language:** It is the native language of computers. It uses only 0s and 1s to represent data and instructions.
 - **Assembly Language:** It corresponds symbolic instructions and executable machine codes and was created to use letters instead of 0s and 1s.
 - **High-level Language:** The programs written in high-level languages are known as source programs and these programs are converted into machine-readable form by using compilers or interpreters.

Generations of Programming Languages

First Generation: Machine Language

- ✓ The first language was binary, also known as machine language, which was used in the earliest computers and machines.
- ✓ Every instruction and data should be written using 0s and 1s.
- ✓ An instruction in machine language consists of two parts. The first part is the command or an operation, which tells the computer what functions are to be performed. The second part of the instruction is the operand, which tells the computer where to find or store the data on which the desired operation is to be performed.



Generations of Programming Languages (Contd.)

Second Generation: Assembly Language

- ✓ It allows the programmer to interact directly with the hardware.
- ✓ This language assigns a mnemonic code to each machine language instruction to make it easier to remember or write.
- ✓ The basic unit of an assembly language program is a line of code.
- ✓ Each line of an assembly language program consists of four columns called fields.
- ✓ The general format of an assembly instruction is:
[Label] <Opcode> <Operands> [; Comment]
- ✓ The assembly language program must be translated into machine code by a separate program called an assembler.



Generations of Programming Languages (Contd.)

Third Generation: High-level Language

- ✓ Languages such as COBOL, FORTRAN, BASIC, and C are examples of 3GLs and are considered high-level languages.
- ✓ Using a high-level language, programs are written in a sequence of statements that impersonates human thinking to solve a problem.
- ✓ Since computers understand only machine language, it is necessary to convert the high-level language programs into machine language codes. This is achieved by using language translators or language processors, which include compilers, interpreters or other routines that accepts statements in one language and produces equivalent statements in machine language.

Generations of Programming Languages (Contd.)

Fourth Generation: 4GL

- ✓ **Fourth generation languages (4GLs)** have simple, English-like syntax rules, commonly used to access databases.
- ✓ Fourth generation languages comprise minimum number of syntax rules. Hence, people who have been trained as programmers can also use such languages to write application programs. This saves time and allows professional programmers to perform more complex tasks. The 4GLs are divided into following three categories:
 - **Query Languages**
 - **Report Generators**
 - **Application Generators**

Generations of Programming Languages (Contd.)

Fifth Generation: Very High-level Languages

- ✓ Fifth generation language actually is a future concept. They are just the conceptual view of what might be the future of programming languages.
- ✓ The computers would be able to accept, interpret, and execute instructions in the native or natural language of the end users.
- ✓ The users will be free from learning any programming language to communicate with the computers.
- ✓ Since these languages are still in their infancy, only a few are currently commercially available. They are closely linked to artificial intelligence and expert systems.

Features of a Good Programming Language

- ✓ Ease of Use
- ✓ Portability
- ✓ Naturalness for the Application
- ✓ Reliability
- ✓ Safety
- ✓ Performance
- ✓ Cost
- ✓ Promote Structured Programming
- ✓ Compact Code
- ✓ Maintainability
- ✓ Reusability
- ✓ Provides Interface to Other Language
- ✓ Concurrency Support
- ✓ Standardisation