

SRINIVAS UNIVERSITY

QUESTION BANK WITH ANSWER FOUNDATION OF INFORMATION TECHNOLOGY

UNIT 3

1. Compilers and Interpreters are themselves?
A. **Programs** B. Driver Software
B. Codes D. Mnemonics
2. Applications are often referred to as _____
A. Datafiles B. System software
C. **Executable files** D. The operating system
3. Prescribed set of well-defined instructions for solving mathematical problems is called
A. A compiler B. A description
C. A code **D. An algorithm**
4. _____ contains specific rules and words that express the logical steps of an algorithm
A. Programming structure **B. Programming language**
C. Syntax D. Logicchart
5. Which among the following is not a programming language?
A. Php B. Perl
C. IEEE D. Java
6. Compiler is related to?
A. **Programming Language** B. DOS
B. Internet D. Database
7. What Word "INTERPRETER" is related to?
A. Address resolution B. IP address ranges
C. Programming Languages D. None of above
8. Firewall is?
A. Hardware **B. Can be Hardware as well as software**
C. Website D. A Software

9. What is an algorithm?

- A. Type of programming language B. Application code
C. **Step-by-step procedure for calculations** D. None of above

10. Which among the following is the lowest form of Computer Language?

- A. Assembly Language B. Perl
C. **Machine Language** D. COBOL

11. Firmware is stored in?

- A. **ROM** B. RAM
C. HardDisk D. Cache

12. What Set of instructions instructs the computer to perform a certain task called?

- A. Assembler B. Debugger
C. **Program** D. Flowchart

13. Which type of software are device drivers

- A. Application software B. **System software**
C. Utility software D. These are types of hardware

14. Which of the following software interacts with the hardware

- A. Application software B. Interpreter
C. **System software** D. Utility program

15. Which of the following is not application software?

- A. Photoshop B. **Winrar**
C. SAGE D. Dream weaver

16. Which among the following is not a low-level language?

- A. Machine Level Language B. Assembly Language
C. **COBOL** D. None of above

17. Programmers who write system software are considered as

- A. **System programmers** B. Analysis programmer
C. Train programmers D. Design programmers

18. Programmer who works in high-level languages and has application understanding are considered

- A. Design programmers B. **Applications programmer**
C. Analysis programmer D. Train programmers

19. Specialized program that allows users to utilize a specific application is classified as

A.Relative programs

B.Application programs

C.Duplicate programs

D.Replicate programs

20. Program which is written originally by the programmer is classified as

A.Object code

B. Machine code

C.Source program

D. Interactive programs

21. Diamond shaped symbol is used in flowcharts to show

A. Decision box

B. Statement box

C.Error box

D. if-statement box

22. Links with another part of the program or connectors in the flowchart are shown in

A. Rhombus

B.Parallelogram

C. circle

D.Trapezoid

23. Part of the algorithm which is repeated a fixed number of times is classified as

A.Iteration

B.Selection

C. Sequence

D. Reverse action

24. In the high-level language Pascal, area is calculated as

A.100 Area=Width*Length

B.100 Area =Width*Length

C. Area: =Width*Length;

D. length 100:area*20width

25. Type of statement written in sequence and repeated until conditions are met is classified as

A.Format

B.Loop

C.Case

D.Condition

LONG ANSWER QUESTIONS

1. Can you explain the different phases of the program development cycle?

Problem Analysis: The problem is analyzed precisely and completely.

Task Analysis: After analyzing the problem, the developer needs to develop various solutions to solve the given problem.

Algorithm Development: After selecting the appropriate solution, an algorithm is developed to depict the basic logic of the selected solution. An algorithm depicts the solution in logical steps



Algorithm Testing: Before converting the algorithms into actual code, it should be checked for accuracy.



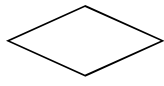
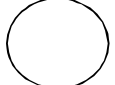
Coding: A program can be written by using computer languages of different levels such as machine, assembly, or high-level languages (HLL).

Testing and Debugging: It is common for the initial program code to contain errors. **Documentation:** Once the program is free from all the errors, the program developers must ensure that the program is supported by suitable documentation. **Implementation:** The program is installed on the end user's machine and the user is also provided with all the essential documents to understand how the program works.

Maintenance and Enhancement: It should be properly maintained by taking care of the changing requirements of its users and the system. The program should be regularly enhanced by adding additional capabilities.

1. Mention and explain the four symbols used in flowcharts

Symbol	SymbolName	Description
	Flow lines	Flow lines are used to connect symbols. These lines indicate the sequence of steps and the direction of flow.
	Terminal	This symbol is used to represent the beginning (start), the termination (end), or the halt in the program logic.
	Input/output	It represents information entering or leaving the system, such as customer orders (input)

		and servicing (output)
	Processing	Process symbol is used for representing arithmetic and data movement instructions. It can represent a single step or an entire sub-process within a larger process.
	Decision	The decision symbol denotes a decision 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 question is yes or no.
	Connector	The connector symbol is used to join different flow lines.

2. Can you write a brief outline about the Categories of software

The software can be categorized as *system software* and *application software*. System software is a generic term for referring to any computer program whose purpose is to help the user to run the computer system, whereas application software employs the capabilities of a computer directly to a task that the user wishes to perform.

System software

System software consists of several programs, which are directly responsible for controlling, integrating, and managing the individual hardware components of a computer system.

System software acts as an interface between the hardware of the computer and the software applications. Some examples of system software are operating systems, device drivers, language translators, and system utilities.

Application software

Software by a user is the application software. Application software may consist of a single program, such as Microsoft's Notepad (for writing and editing simple text), or a collection of programs, which work together to accomplish a task such as a database management software. Application software is dependent on system software. System software (like an operating system) acts as an interface between the user and the computer hardware, while application software performs specific tasks.

Some of the most commonly used application software, are Word processors, spreadsheets, Image Editors, and Database Management systems.

3. Identify different Benefits of flowcharts.

The reasons for using flowcharts as a problem-solving tool are given below:

Makes Logic Clear: The main advantage of using a flowchart to plan a task is that it provides a pictorial representation of the task, which makes the logic easier to follow.

Effective Analysis: With the help of a flowchart, the problem can be analyzed effectively. This is because the analyzing duties of the programmers can be delegated to other persons, who may or may not know the programming techniques, as they have a broad idea about the logic.

Useful in Coding: Once the flowcharts are ready, the programmers can plan the coding process effectively as they know where to begin and where to end, making sure that no steps are omitted.

Proper Testing and Debugging: A flowchart helps in detecting the errors in a program, as the developers know exactly what the logic should do. Developers can test various data for a process.

Appropriate Documentation: Flowcharts serve as a good program documentation tool. They can take the help of the program documentation to know what the program does and how to use the program.

4. Summarize the drawbacks of flowchart

Complex: The major disadvantage of using flowcharts is that when a program is very large, the flowcharts may continue for many pages, making them hard to follow.

Costly: If flowcharts are to be drawn for a huge application, the time and the cost factor of program development may get out of proportion, making it costly.

Difficult to Modify: Any changes or modification to a flowchart usually requires redrawing the entire logic again, and redrawing a complex flowchart is not a simple task.

No Update: Usually programs are updated regularly. However, the corresponding update of flowcharts may not take place, especially in the case of large programs.

5. How would you Explain the following terminologies:

- **Freeware**
- **Shareware**
- **Open-Source Software**
- **Proprietary Software**

Freeware: The term freeware is commonly used for copyrighted software that is given for free by its owner. Though it is available for free, the owner retains the copyright, which means that a user does not have the right to modify anything in the software.

Shareware: Shareware is software that is distributed free on a trial basis. It allows people to redistribute copies for a limited period. Anyone who continues to use a copy is required to pay a license fee. Therefore, the free use of the software is usually limited to a certain period.

Open-Source Software: Open-source software is created by generous programmers and released into the public domain for public use.

Proprietary Software: In general, proprietary describes a technology or product that is owned exclusively by a single company that carefully guards knowledge about the technology or the product's internal working.

6. Can you write an algorithm to find the largest of the three numbers?

Algorithm 1.

Step 1: Start

Step 2: Read the three numbers A, B, C

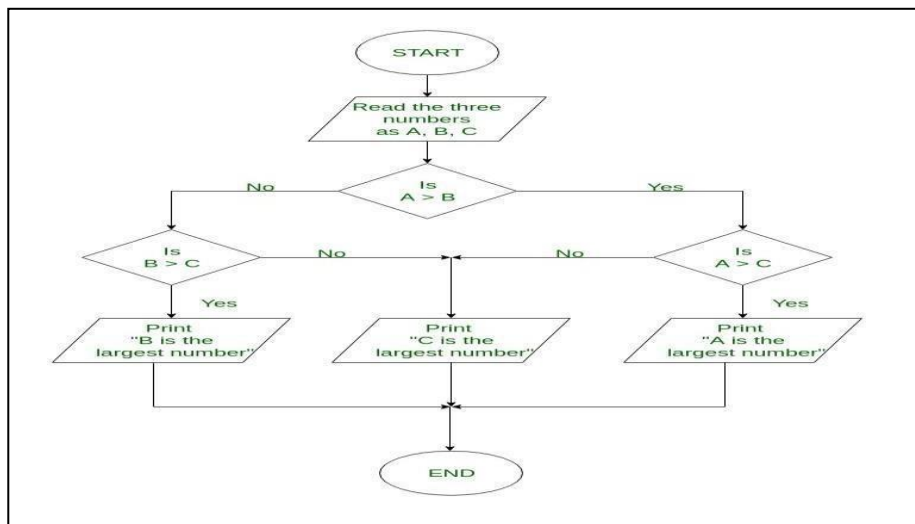
Step 3: Compare A and B. If A is greater perform step 4 else perform Step 5.

Step 4: Compare A and C. If A is greater, output "A is greatest" else Output "C is greatest". Perform step 6.

Step 5: Compare B and C. If B is greater, output "B is greatest" else output "C is greatest".

Step 6: Stop.

7. Draw a flowchart to find the largest of the three numbers.



8. Explain the features of good programming language.

Ease of use: The language should be easy in writing codes for the program and execute them. Simplicity helps in the readability of the language.

Portability: The language should support the construction of code in a way that it could be distributed across multiple platforms(OS).

Generality: Most high-level languages allow the writing of a wide variety of programs, thus relieving the programmer of the need to become an expert in many diverse languages.

Brevity: Language should have the ability to implement the algorithm with less amount of code.

Error checking: Being human, a programmer is likely to make many mistakes in the development of a computer program. Many high-level languages enforce a great deal of error checking both at compile-time and run-time.

Cost: It includes several costs such as Program execution and translation cost, program creation, testing and usage cost, and program maintenance costs.

Familiar notation: A language should have familiar notation, so it can be understood by most programmers.

Quick translation: It should admit quick translation.

Reusability: Code is reusable when it is independent of other codes.

Modularity: It is desirable that programs can be developed in the language as a collection of separately compiled modules, with appropriate mechanisms for ensuring self-consistency between these modules.

Widely available: Language should be widely available and it should be possible to provide translators for all the major machines and all the major operating systems.

Performance: Now a day's hardware has become quite fast. Hence the application developed using a good language should tap the maximum resources of the available hardware power in terms of speed and memory efficiency

9. Define Software, Illustrate the relationship between hardware and software

A computer system consists of hardware, the electronic devices that are capable of computing and manipulating information, and software that carries out predefined tasks to complete a given job. The sets of instructions, which control the sequence of operations, are known as programs, and collectively programs are called software.

Software is a generic term for an organized collection of computer data and instructions. Software tells the computer what to do and how to do it. For example, software instructs the hardware what to display on the user's screen, what kind of input to take from the user, and what kinds of output to generate.

A computer needs to be instructed to perform any task. These instructions are given in the form of computer programs, which are written in computer programming languages.

RELATIONSHIP BETWEEN SOFTWARE AND HARDWARE

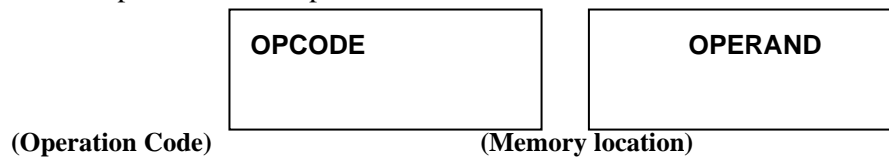
Software refers to the computer programs that are loaded into a computer system, and hardware refers to all the visible devices, which are assembled to build a computer system

Hardware is like a car without a driver; one needs both to get the work done. Software is a set of instructions that tells the hardware *what* to do and *how* to perform the requested action. Thus, hardware and software share a special relationship. If hardware is the "heart" of a computer

system software is its "soul." Both are complementary to each other.

10. Illustrate machine-level 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 0's and 1's. Machine language is also known as the computer's "native" language as this system of codes is directly understood by the computer. Instruction in machine language consists of two parts. The first part is an operation that tells the computers 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.



11. Briefly give an outline of Assembly-level language.

This language assigns a mnemonic code to each machine language instruction to make it easier to remember or write. An assembly language provides a mnemonic instruction, usually 3 letters long, corresponding to each machine instruction. The letters are usually indicating what the instruction does. For example, ADD is used to perform an addition operation, SUB for subtraction, and so on. Assembly languages make it easier for humans to remember how to write instructions to the computer.

Each line of the assembly language program consists of 4 columns called fields. The general format of an assembly instruction is as follows.

[Label] <opcode><operands> [; comment]

[...] brackets indicate that the enclosed specifications may or may not appear in a statement. If a label is specified, it is associated as a symbolic name with the machine words generated for the assembly statement. If multiple operands are used, each of them is separated by a comma. The text after the semicolon

(;) is just comments. Comments are not a part of the actual program but are used just for reference purposes, ie, to specify what the statement will do.

Label	Opcode	Operands	Comments
-------	--------	----------	----------

BEGIN	to	ADD	A, B; ADD B to A
-------	----	-----	------------------

Assembler The assembly language must be translated into machine code by a separate program called an assembler

12. What is high-level language explain the translators used.

Languages such as COBOL, BASIC, and C are examples of 3GLS and are considered HLL. HLL is similar to the English language. Programs written using these languages can be machine-independent. A single high-level statement can substitute several instructions in the machine or assembly language. In high-level level language, programs are written in a sequence of statements to solve a problem. Forex, the following BASIC code will calculate the sum of 2 numbers.

```
LET X=10 LET Y=20
```


LET SUM=X+Y

PRINT SUM

The first two statements store 10 in variable x and 20 in variable y. The third statement again creates a variable named sum, which will store the summation of the x and y values. Finally, the output is printed on the screen.

Translating HLL into machine language

Computers understand only machine language, so convert the HLL into machine language code by using language translators known as compilers or interpreters.

Compiler A compiler translates a program into another program, known as the target language.

Interpreter An interpreter translates a statement in a program and executes the statement immediately, before translating the next source language statement.