BT-205 Basic Computer Engineering Unit-II Notes

UNIT-II TOPICS

- Introduction to Algorithms
- Complexities and Flowchart
- Introduction to Programming
- Categories of Programming Languages
- Program Design
- Programming Paradigms
- OOP Concepts
- POP vs OOP
- Introduction to C++
- Character Set and Tokens
- Operators and Expressions
- Statements and Control Structures
- Input and Output Operations
- Arrays and Functions

1. ALGORITHM

Algorithm is a step-by-step procedure to solve a problem.

Characteristics of Algorithm:
- Finite steps
- Well-defined instructions
- Input and output
- Efficiency

Example:
Algorithm to add two numbers:
Step 1: Start
Step 2: Read A and B
Step 3: Sum = A + B
Step 4: Print Sum
Step 5: Stop

2. FLOWCHART

Flowchart is graphical representation of algorithm.

Common Symbols:
- Oval → Start/Stop
- Rectangle → Process
- Diamond → Decision
- Parallelogram → Input/Output

Advantages:
- Easy to understand
- Helps in debugging
- Better program planning

3. COMPLEXITY

Complexity measures efficiency of algorithm.

Types:
- Time Complexity
- Space Complexity

4. INTRODUCTION TO PROGRAMMING

Programming is process of writing instructions for computer.

Steps in Programming:
• Problem analysis
• Algorithm design
• Coding
• Testing
• Debugging

5. CATEGORIES OF PROGRAMMING LANGUAGES

(a) Low Level Language
• Machine language
• Assembly language

(b) High Level Language
• C
• C++
• Java
• Python

6. PROGRAM DESIGN

Program design is planning structure and logic of program before coding.

Tools:
• Algorithm
• Flowchart
• Pseudocode

7. PROGRAMMING PARADIGMS

Programming paradigms are styles of programming.

Types:
• Procedural Programming
• Object Oriented Programming
• Functional Programming

8. OBJECT ORIENTED PROGRAMMING (OOP)

OOP is programming approach based on objects and classes.

Main Concepts of OOP:
• Class
• Object
• Encapsulation
• Inheritance
• Polymorphism
• Abstraction

Advantages of OOP:
• Code reusability
• Data security
• Easy maintenance

9. POP VS OOP

Procedure Oriented Programming:
• Focus on functions

• Less security
• Example: C language

Object Oriented Programming:
• Focus on objects
• More security
• Example: C++

10. INTRODUCTION TO C++

C++ is object oriented programming language developed by Bjarne Stroustrup.

Features:
• Fast execution
• Supports OOP
• Portable language

11. CHARACTER SET

Character set includes letters, digits and symbols used in C++.

Examples:
• Alphabets
• Numbers
• Special symbols

12. TOKENS

Tokens are smallest units in C++ program.

Types of Tokens:
• Keywords
• Identifiers
• Constants
• Operators
• Strings

13. PRECEDENCE AND ASSOCIATIVITY

Precedence defines priority of operators.
Associativity defines order of evaluation.

14. PROGRAM STRUCTURE

Basic structure of C++ program:

```
#include <iostream>
using namespace std;
int main()
{
cout << "Hello";
return 0;
}
```

15. DATA TYPES

Data type specifies type of data.

Examples:
• int
• float
• char

• double

16. VARIABLES

Variables are memory locations used to store data.

Example:
int a = 10;

17. OPERATORS

Operators perform operations on operands.

Types:
• Arithmetic Operators
• Relational Operators
• Logical Operators
• Assignment Operators

18. EXPRESSIONS

Expression is combination of variables, constants and operators.

Example:
a + b * c

19. STATEMENTS

Statements are instructions executed by program.

Types:
• Declaration statement
• Assignment statement
• Conditional statement

20. CONTROL STRUCTURES

Control structures control flow of execution.

Types:
• if statement
• if-else
• switch
• loops (for, while, do-while)

21. INPUT AND OUTPUT OPERATIONS

C++ uses cin and cout for I/O operations.

Example:
cin >> a;
cout << a;

22. ARRAY

Array stores collection of similar data items.

Example:
int arr[5];

23. FUNCTIONS

Function is block of code performing specific task.

Advantages:
• Code reusability
• Better readability

Example:
```
int add(int a,int b)
{
return a+b;
}
```

MOST IMPORTANT 14 MARK QUESTIONS

1. Explain algorithm and flowchart with examples.

2. Explain programming paradigms and characteristics of OOP.

3. Differentiate POP and OOP.

4. Explain structure of C++ program and tokens.

5. Explain operators, expressions and control structures in C++.

6. Explain arrays and functions with examples.

7. Explain categories of programming languages.

8. Explain precedence and associativity in C++.

IMPORTANT 7 MARK QUESTIONS

1. Define algorithm.

2. Explain flowchart symbols.

3. Explain OOP concepts.

4. Explain data types and variables.

5. Explain loops in C++.

6. Explain functions in C++.

7. Explain arrays with examples.

8. Explain tokens in C++.

EXAM TIPS

• Practice flowchart drawing regularly.
• Learn OOP concepts carefully.
• Revise operators and loops properly.
• Practice small C++ programs daily.
• Focus on POP vs OOP differences.