XI COMPUTER – NON C++ MATERIAL

VERY IMPORTANT NOTE

Dear Student, this material is meant only for slow learners to give an idea of questions pattern, collected from 3 old question papers only. Students are advised to read entire syllabus.

XI COMPUTER – PROGRAMMING METHODOLOGY

UNIT 2: PROGRAMMING METHODOLOGY - SYLLABUS

General Concepts: Modular Approach, Clarity and Simplicity of Expressions, Use of proper names for Identifiers, Comments, Indentation; Documentation and Program Maintenance; Running and Debugging programs, Syntax Errors, Run-Time Errors, Logical Errors
Problem Solving Methodologies: Understanding of the problem, Solution for the problem, Identifying minimum number of inputs required for output, Writing code to optimizing execution time and memory storage, step by step solution for the problem, breaking down solution into simple steps (modular approach), Identification of arithmetic and logical operations required for solution;
Control Structure- Conditional control and looping (finite and infinite).
Problem Solving: Introduction to Algorithms/Flowcharts.

MATERIAL

The software designing involves mainly two things – program structure and program representation. **Program structure** refers to how a program should be. The program structure is decided using top-down approach or any other popular approach. The main task is divided into clear logical subtasks.

The **program representation** refers to its presentation style so that the program becomes more readable and presentable.

STYLISTIC GUIDELINES:

(i) Modular approach to programming : Always break a big program into smaller subcomponents called modules – this is called modular approach to programming. Each module is composed of an independent or self-contained set of instructions. Each module is designed to perform a specific task in the overall program. The advantage of modular programming is the ability to write and test each module independently and in some cases reuse modules in other programs.

(ii) Meaningful names for identifiers: Assign meaningful names for all identifiers ie variables, arrays, functions, etc. Do not use similar looking names.

(iii) Ensure clarity of Expression.

(iv) Use comments

(v) Use indentation - Indentation makes the statements clear and readable. Indentation is not mandatory in C++.
Indentation is used to highlight nesting of groups of control statements.

(vi) Insert blank lines and blank spaces – Insertion of blank lines and blank spaces at appropriate spaces also enhances the program readability.

COMMENTS: Comments are the pieces of code that the compiler does not execute. The purpose of comments is to make program more understandable. They provide internal documentation of a program.
There are **two ways to insert comments** in C++ program:

(i) Single line comments:
Starts with //. The compiler ignores everything following //after in that same line.
Eg: c =a + b; //This statement is used to add two numbers.

(ii) Multiline or block comments: Starts with /* and ends with */. Everything that is in between /* and */ will be ignored by the compiler.

Types of comments:

(a)Prologues: Comments in the beginning of a program/functions that indicates the purpose of the program/function.

(b) Explanatory comments: Insert explanatory information wherever applicable since they explain the role and purpose of other identifiers (variables) and statement(s).

(c) Block comments : Insert block comments to identify the purpose of block statements/ { ---- } pairs.

Example Program:

```
//This program checks given number is prime or not - Prologue
#include<iostream.h>
#include<conio.h>
void main( )
{ int i,mul,n; //mul variable for number of factors – explanatory comment.
  i=1; mul=0;
  clrscr( );
  cout<<"\n Enter any number: ";
  cin>>n;
  while(i<=n)          // this while loop finds factors of n – block comment
  {  if(n%i==0)        // if statement indented inside while loop
            mul++;
     i++;
  }
  if(mul==2)
    cout<<n<<" is a prime number";
  else
    cout<<n<<" is not a prime number";
  getch( );
}
```

CHARACTERISTICS OF A GOOD PROGRAM:

(i) Effective and efficient (ii) User Friendly
(iii) Self-Documenting code (iv) Reliable (v) Portable.

STATEMENT FORMATTING STYLE:

(i) Free formatting style : Number of statements can be written in one line using a statement separator (;).

Ex:

```
#include<iostream.h>
void main( ) { int a=10,b=20,c; c=a+b;cout<<"Addition ="<<c; }
```

It is not readable and understandable.

(ii) Prettyprinting: When the program formatting is done to make a program more readable, it is called prettyprinting.

Ex:

```
#include<iostream.h>
void main( )
{ int a=10,b=20,c;
  c=a+b;
  cout<<"Addition ="<<c;
}
```

DIFFERENT TYPES OF ERRORS: Error is also called as bug.

1.Syntax Errors: Syntax errors occur when rules of a programming language are misused. **Syntax** refers to formal rules governing the construction of valid statements in a language.

Eg: int a,b //Did not keep ; (semicolon) at the end of statement.

2.Semantics Error: Semantic errors occur when statements are not meaningful. **Semantics** refers to the set of rules which give the meaning of a statement.

Eg: X * Y = Z;

(Siva plays Guitar is Syntactically and Semantically correct but Guitar plays Siva is Syntactically correct but Semantically incorrect).
(Syntax errors and Semantics together are called as compile time errors)

3.Type Errors: If a function is given wrong type of data, type error will occurs. If the integer argument was expected but actually string was given in place of it, it is a type error.

4.Run-time Errors(Execution errors): A run-time error is that occurs during the execution of a program.

Eg: 1) Division by zero. C= a/b; if b value given by user is 0.
2**)** If a program is trying to open a file which does not exist or it could not be opened, it results into an execution error.

5.Logical Errors: A logical error is that error which causes a program to produce incorrect or undesired output.

Eg: If you want to add two numbers but you gave '*' instead of '+' . ie c = a * b;

STAGES OF PROGRAM DEVELOPMENT:

(i) Crack the problem
(ii) Code the algorithm
(iii) Compile the program
(iv) Execute the program

Robustness: The ability of a program to recover following an error and to continue operating within its environment, is called robustness.

Guard code: The code which can handle exceptional data errors and operational errors is called guard code.

PROBLEM SOLVING METHODOLOGY AND TECHNIQUES:

The steps to create a working program :

(1) Understand the problem well: Under this step we must understand the problem well so as to figure out "what is required or desired out of the proposed solution?"

(2) Analyse the problem:

Identify minimum number of inputs required for output

Identify processing components.

(3) Design programs: The solution to the problem is planned. Plan logical sequence of precise steps that solve the problem ie algorithm. Then draw flowchart using these steps.

In top-down design(divide and conquer), major steps are listed. Then break down each step(main routine) into smaller steps (subroutines or sub programs). These again can break down in to smaller steps. Do the same until each one performs a single function or task.

Design the final program by : Deciding step by step solution then breaking down solution into simple steps.

(4) Code Programs using appropriate control structures: Writing the program. Translate the algorithm into programming language.

(5) Test and debug programs: Testing is the process of finding errors in a program. Debugging is the process of correcting errors found during the testing process. The goal of program testing is to ensure that the program runs correctly and is error free.

(6) Complete the documentation: Documentation allow another person or the programmer at later date to understand the program. Internal documentation consists of statements in the program that are not executed, but point out the purposes of various parts of the program. Documentation might also consists of a detailed description of what the program does and how to use the program.

Documentation may includes an instruction manual.

Implement your code: After testing and documentation, implementation your program for actual use on site. Now, the real users can use your programs.

(7) Maintain Programs: Involves modifying the programs to remove previously undetected errors, to enhance the program with different features or functionality, or keep the program up-to-date as government regulations or company policies change, etc.

/* ABOVE ANSWER FOR 2 MARKS QUESTION */

Write down the steps to be followed while running a program.

A. **Steps to create a working program:**

(1) Understand the problem well.

(2) Analyze the problem to

* Identify minimum number of inputs required for output.
* Identify processing components.

(3) Design the program by

* Deciding step by step solution.
* Breaking down solution into simple steps.

(4) Code the program by

* Identifying arithmetic and logical operations required for solutions.
* Using appropriate control structures such as conditional or looping control structure.

(5) Test and Debug your program by

* Finding errors in it.
* Rectifying the errors.

(6) Complete your documentation.

(7) Maintain your program.

/* ABOVE ANSWER FOR 2 MARKS QUESTION COMPLETED */

DOCUMENTATION: Documentation refers to written description, specifications, design, code and comment, internal and external to a program, which make a program more understandable, readable and more easily modifiable.

Documentation must describe the 'who', 'what', 'when', 'where', and 'how' of each system application.

Modules make information easily accessible to the specific user for which they were prepared and they reduce the costs of production and maintenance. The documentation modules are generally referred to as manuals.

The aggregation of modules and their details would depend upon :

(i) Complexity of system

(ii) Technical sophistication of user

(iii) People involved in development and use

(iv) Expected life of documentation

Modules of documentation

Manual	Description
User's Manual	Describes the software product's features and explains the instructions and procedure to use the software product.
Input Preparation Manual	Describes what all forms of input is required to work on the software product and how input can be prepared for the software.
Operations Manual	Describes what all operations are taking place and what their interdependencies are.
Equipment Manual	Describes the equipment required and supported.
Programmer Manual	Describes the code, coding conventions followed in the development of the software.
Systems Manual	Includes all of the documents describing the system itself from the requirements specification to the final acceptance test plan.
Standards Manual	Describes the standards such as process standards, products standards, interchange standards, followed for high quality product and documentation.

Uses of Documentation:

(i) Facilitates communication about an application between the technical development personnel and the non-technical users.

(ii) Useful for technical personnel not only during development, but also during operations and normal maintenance.

(iii) Essential during abnormal or urgent maintenance.

(iv) Very helpful in initiating and training.

(v) Enables troubleshooting when the application system breaks down.

(vi) Helps corporate management understand the system sufficiently to make decisions and to appreciate its financial and resource implications.

(vii) Essential for both internal and external control.

(viii) Ensures that all commitments and expectations are on record.

(ix) Provides a package that will contain the information needed to change the system in case environment or management's needs alter.

(x) Prevents system dislocation and cost that might otherwise occur if knowledge of the system were centered in a few individuals who resigned, relocated, or were subsequently reassigned other duties.

PROGRAM MAINTENANCE: Program maintenance refers to the modification of a program, after it has been completed, in order to meet changing requirements or to take care of errors that show up. **Types of maintenance:**

(i) Corrective Maintenance: Maintenance done to correct the errors after it out program put in operation, is called as corrective maintenance.

(ii) Adaptive Maintenance: Maintenance done for changes in the environment ie to incorporate new rules, changing needs time to time, etc

(iii) Preventive Maintenance: If possible errors could be anticipated before they actually occur, the maintenance could be done to avoid them and the system down time can be saved. Because this type of maintenance aims at preventing errors, it is called preventive maintenance.

(iv) Perfective Maintenance: No technology is permanent. Every year, new technologies come with new features, new facilities. If the existing system is maintained to keep attuned with new features, new facilities, new capabilities, it is said to be perfective.maintenance.

ALGORITHM: The step by step solution to solve any problem is called as algorithm. An algorithm is composed of a finite set of steps, each of which may require one or more operations. (**Or)**

The logical sequence of precise steps that solve a given problem, is called Algorithm.

Each operation must be definite.

Each operation must be effective.

Each operation must be finite.

The algorithm should terminate after a finite number of operations.

Ex: Algorithm for addition of two numbers

Step 1 : Start
Step 2 : Declare a,b,c
Step 3 : Write (Output/Print) 'Enter any two numbers'
Step 4 : Read (Input) a,b
Step 5 : c ← a + b
Step 6 : Write 'The addition of two numbers = ', c
Step 7 : Stop

Ex 2.Write an algorithm to find area of a rectangle.

A) Step 1: Start
Step 2: Take length and breadth and store them as L and B?
Step 3: Multiply by L and B and store it in area
Step 4: Print area
Step 5: Stop

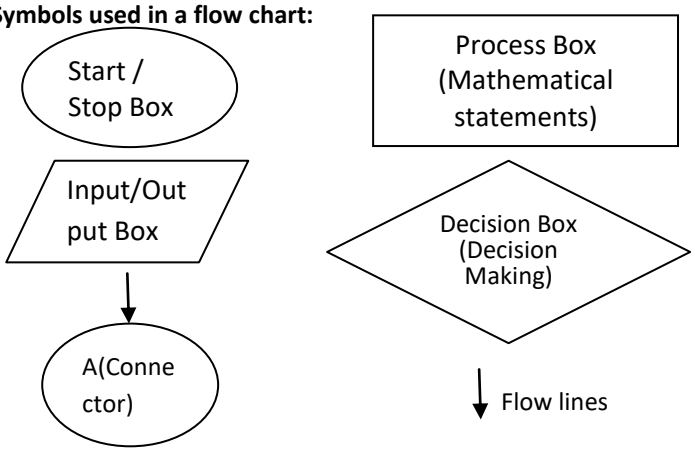
Ex 3.Write an algorithm to find x^y.

A) Step1: Start
Step2: Declare x, y as integers
Step3: Output "Enter the value of x and y ie base and exponent"
Step 4:Read x,y
Step 5:Output "The result of the expression =", pow(x,y)
Step 6: Stop

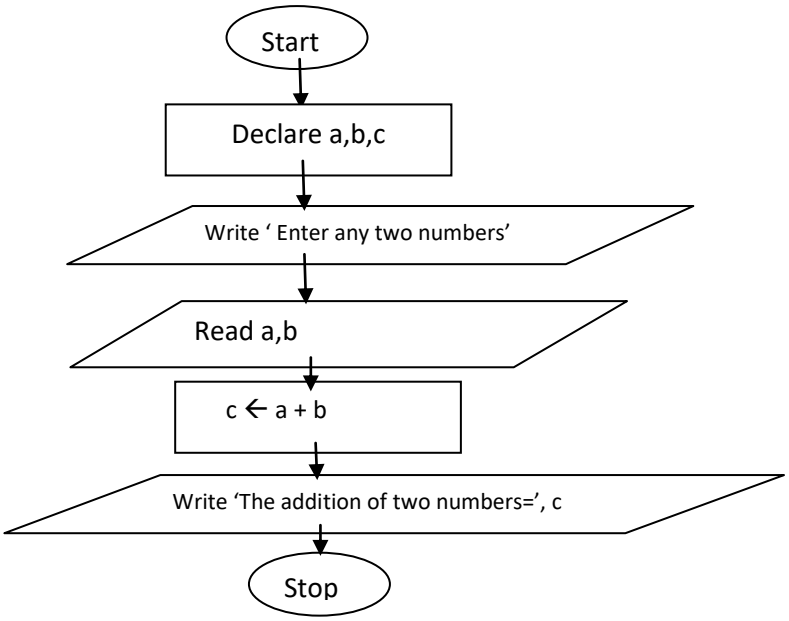
FLOW CHART:

A flowchart is a pictorial representation of step by step solution of a problem.

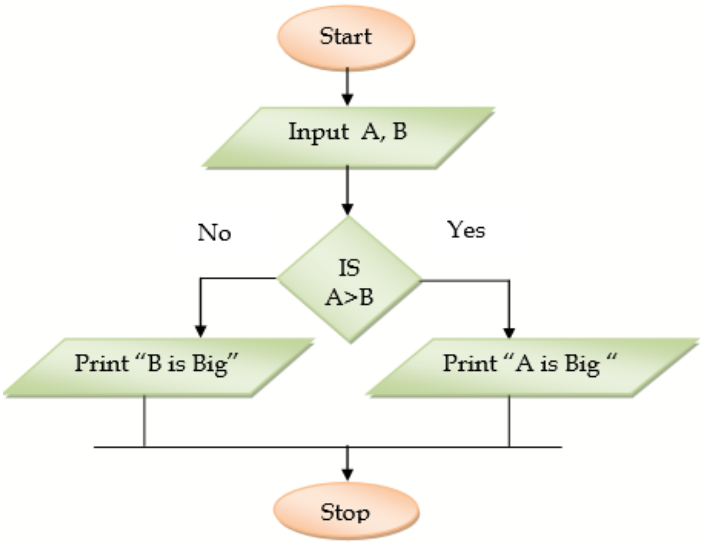
Symbols used in a flow chart:



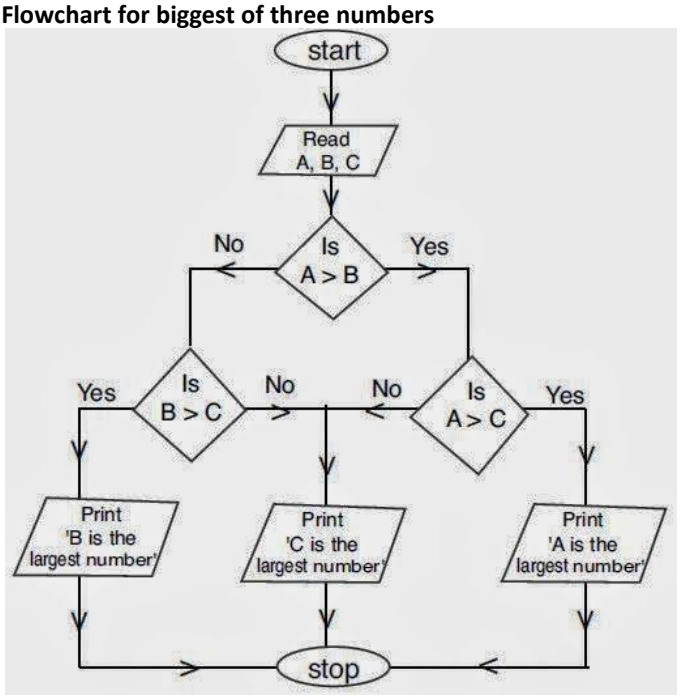
Ex: Flowchart for addition of two numbers



Flowchart to find bigger number between two numbers.



Flowchart for biggest of three numbers



WRITING STRUCTURED PROGRAMS:

While writing algorithms, certain simple rules of style should be applied:
(i) The meaning of all variables and constants should be defined.
(ii) The flow of the program should generally be forward except for normal looping or unavailable instances.
(iii) The code should be indented so that computational units of program text can more easily be identified.
(iv) Documentation should be short, but meaningful. Use comments wherever possible without cluttering the program code.

GUIDELINES FOR EFFICIENT CODING:

(i) Reduce repeated expressions.
(ii) Try to use pure integer expressions or real expressions rather than having mixed expressions.
(iii) Try to combine loops
(iv) Do not put unnecessary statements inside loop
(v) Try to save on comparisons.

PREVIOUSLY GIVEN QUESTIONS

1. What is the difference between semantics error and syntax error? Give an example of each. 3
2. Differentiate between the logical errors and runtime errors with one example each? 2
3. How many types of errors are there? Discuss each with example. 4
4. Explain all the types of errors in C++. Give example of each type of error. 3
5. Differentiate between run time and compile time errors with an example. 3
6. Write down the steps to be followed while running a program. 2
7. What are the steps to create a working program? 2
8. Write all the steps of Problem Solving Methodology. 4
9. What is the use of comments in a program and its types? 2

10. What is the role of comments and indentation in a program? 4
11. What are the characteristics of a good program? 2
12. Write the characteristics of a good program. 2
13. An Error in a program is known as **bug**. 1
14. The process of correcting errors is called as **debugging** 1
15. Write an algorithm to find x^y. 2
16. Write an algorithm to find area of a rectangle. 2
17. Draw the symbol used in the flow chart to read the values? 1
18. Draw a flow chart to find the sum of first 50 even numbers. 4
19. Draw a flowchart to find bigger number between two numbers. 2
20. What do you understand by Guard Code? 1
21. Draw the symbol used in the flow chart for mathematical calculations (process) 1
22. What is documentation? What are the uses of documentation? 3
23. What is prettyprinting? 1
24. What is maintenance? Mention the types of maintenance. 2
25. What is a prologue? 1
26. What is robustness? 1
27. Explain algorithm and flow chart. 2
28. Differentiate between syntax and semantics error. 2

XI COMPUTER – GENERAL OOP CONCEPTS

Evaluation of Programming Languages:
Computer languages are of 2 types.
(1) Low level languages. (2) High level languages.

Low level languages: These are of two types.
(A) Machine Language. (B) Assembly Language.

Machine Language: In machine language, instructions are written in 0 and 1 only. It is the only language that computer can execute directly. As it consists binary digits(0 and 1) only, it is also called as binary language. As the computer can execute machine language programs directly, program execution takes less time. But Program development, modification and debugging (finding and rectifying errors) is very hard. The machine language is called as first generation of computer languages.

Assembly Language: Assembly language programs consists mnemonics (commands for various operations) instead of 0's and 1's. So for the programmer, program development, modification and debugging is easier than machine language programs. But these assembly language programs cannot understood by the computer directly.

Assembler is a program, which converts assembly language program into machine language program. Eg: 8085, 8086, etc.
Assembly Language Program → Assembler → Machine Language Program
Assembly language is called as second generation of computer languages. Low level languages are machine oriented and require knowledge of computer circuitry.

High Level Languages: These are of two types.
(A) Procedure Oriented Programming Languages.
(B) Object Oriented Programming Languages.
High level languages are English like languages. So it is very easy for the programmer to develop, modify and debug high level language programs. But these high level language programs are cannot directly understood by the computer. A compiler or an interpreter will convert high level language programs into machine language programs.
Eg: BASIC, COBOL, C, C++, Java, Visual Basic

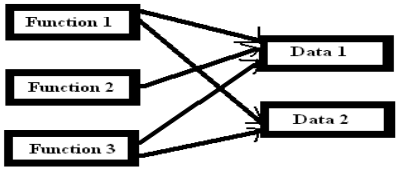
Compiler:
(a)It converts a high level language program into machine language, entire HLL program in one go, and reports all the errors of the program along with the line numbers.
(b)Whenever the program is compiled, then only the compiler should available in memory. High Level Language Program → Compiler → Machine Language Program

Interpreter:
(a)This converts a high level language program into machine language by converting and executing it line by line.
(b)If there is any error in any line, it reports it at the same time and program execution cannot resume until the error is rectified.
(c)Interpreter must always be present in the memory every time the program is executed. High Level Language Program→ Interpreter→Machine Language Program

1.2 Programming Paradigm:
A programming paradigm defines the methodology of designing and implementing programs.

(A) Procedure Oriented Programming Languages(POP):
(i)A program in a procedural language is a list of instructions where each statement tells the computer to do something.
(ii)The focus is on the processing.
(iii)There is no security for data.

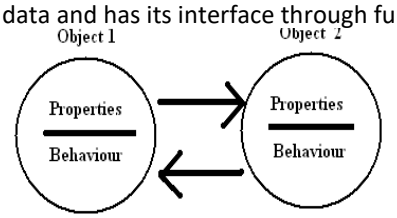
Limitations of Procedural Programming Paradigm:
(i) Emphasis on algorithm (or procedure) rather than data.
(ii) Change in a data type being processed needs to be propagated to all the functions that use the same data type. This is a frustrating and time-consuming process.
(iii) The procedural programming paradigm does not model real world very well.
Ex.,of Procedural Programming Languages: BASIC, COBOL, C, Foxpro.



All data is openly available to all functions in the program.

(B)Object Oriented Programming Language(OOP):
In OOP, we can represent real world object into software terms. Every real world object have its own characteristics (properties) and behavior.
In OOP, Characteristics will be represented through member variables, behaviour will be represented through member functions.
In OOP, the behaviour of an object will be represented through class.
Then we can create number of objects of that class.

Class is an abstract idea, object is a physical thing.
Car is a class, Maruti Car is an object.
Pen is a class, Cello Pen is an object.
(i) The object oriented approach views a problem in terms of objects involved rather than procedure for doing it.
(ii) In object oriented programming, object represents an entity that can store data and has its interface through functions.



Data and functions enclosed within objects. New objects communicate with one another. Ex.,of Object Oriented Programming Languages: Simula67, Java, etc.,

Advantages of OOP:
(i) It models real world well.
(ii) With OOP, programs are easy to understand.
(iii) OOP offers classes reusability. Already created classes can be reused without having to write them again.
(iv) OOP facilitates Quick Development.
(v) With OOP, programs are easier to test, manage and maintain.

Procedure Oriented Programming	Object Oriented Programming
A program in a procedural language is a list of instructions where each statement tells the computer to do something. The focus is on the processing. There is no much security for the data.	The object oriented approach views a problem in terms of objects involved rather than procedure for doing it. In object oriented programming, object represents an entity that can store data and has its interface through functions.

All data is openly available to all functions in the program	Data and functions enclosed within objects.
	OOP is so closer to the real life. And also it consists many features like 1.Data Abstraction 2.Data Encapsulation 3 .Modularity 4.Inheritance 5.Polymorphism 6. Data hiding.

(C)Object Based Programming Language: These programming languages have the features of OOP except inheritance and polymorphism.

Object Oriented Programming Concepts:

The following are some important OOP concepts.

1. Class
2. Object
3 .Data Abstraction
4 .Data Encapsulation
5 .Polymorphism
6. Inheritance
7. Modularity

1. Class: A class is a group of objects that share common properties and relationships. In OOP, the behavior of an object will be represented through class.

Then we can create number of objects of that class.

2. Object: Object is an identifiable entity with some characteristics and behavior.

In OOP, object represents an entity that can store data and has its interface through functions.

Class is an abstract idea, object is a physical thing.

Car is a class, Maruti Car is an object.

Pen is a class, Cello Pen is an object.

3. Abstraction: Abstraction refers to the act of representing essential features without including the background details or explanations.

Example: (1) You are driving a car. You only know the essential features to drive a car ie gear handling, steering handling, use of clutch, accelerator, brakes, etc., But while driving do you get into internal details of car like wiring, motor working etc? You just change the gears or apply the brakes etc. What is happening inside is hidden from you. This is abstraction where you only know the essential things to drive a car without including the background details or explanations.

(2) Example is student data: If you take a student details, he will have his name, roll number, class, section, various marks, total, average, father name, address, date of birth, cell number, etc. But when we think of his marks details, we consider only roll number, name, class, section, various marks, total and average only. This is abstraction where you only know the essential things of a student for his marks details without including all the details.

(**Eg:** If you talk about student details, complete details are his rollno, admissionno, name, m1, m2, m3, total, avg, grade, fathername, address, pin, etc……

But abstraction about his marks are…

Rollno,admissionno,name,m1,m2,m3,total,avg,grade.)

4. Encapsulation: The wrapping up of data and functions (that operate on the data) into a single unit (called class) is known as Encapsulation. Encapsulation is a feature of hiding the complexity of the data. Consider the example of a car. If we want to stop the car, we just apply the brake and we don't need the mechanism of the braking system. The mechanism behind braking is encapsulated. Encapsulation hides the implementation details of an object and hence also called as data hiding.

Abstraction shows only the relevant details, where as Encapsulation hides the irrelevant details.

A class binds together data and its functions under one unit thereby enforcing encapsulation as encapsulation means wrapping up data and associated functions together into a single unit.

5. Modularity: The act of partitioning a program into individual components (modules) is called modularity. A **module** is a separate unit in itself. It can be compiled separately though it has connections with other modules. Modules work hand in hand in order to achieve the program's goal.

6. Inheritance: Inheritance is the capability of one class of things to inherit capabilities or properties from another class. The class, whose properties are inherited, is called **Base class** or **Super class** and the class that inherits these properties, is called **Derived class** or **Sub class.** (Inheritance is closer to the real world. We will get our genetics, nature from our parents.)

7. Polymorphism: Polymorphism means many forms. Polymorphism is the ability for a message or data to be processed in more than one form.

Polymorphism is the concept that supports the capability of an object of a class to behave differently in response to a message or action.

Example: Consider a bird, see is one of the behaviour of the bird. If bird tried to see in day time, it can able to see. But if bird tried to see in night time, it cannot see. For the same bird, for the same behaviour ie see, different response. This is called as polymorphism.

Data Hiding: Keeping the data in private visibility mode of the class to prevent it from accidental change is known as Data Hiding.

```
class Computer
{  char CPU[10];int RAM;          //Data Hiding
public:                           //Data Encapsulation
void STOCK();
void SHOW();
};
```

GENERAL OOP CONCEPTS

1. What is a paradigm? 1
2. Differentiate between object oriented programming and procedural oriented programming with the help of examples of each. 2
3. Write any 2 uses of OOPs. 2
4. What is data abstraction? Explain the concept with the help of an example. 2
5. What do you mean by Abstraction and Encapsulation? How are these two terms interrelated? 4
6. What is encapsulation? 1
7. What is polymorphism? 2

UNIT 1 : CHAPTER 1 TO 4 :: SYLLABUS

Unit 1: Computer Fundamentals

Classification of computers: Basics of computer and its operation; Functional Components and their interconnections, concept of Booting.

Software concepts: Types of Software – System Software, Utility Software and Application Software

System Software: Operating System, Compiler, Interpreter and Assembler;

Operating System: Need for Operating System, Functions of Operating System (Processor Management, Memory Management, File Management and Device Management), Types of Operating System- Interactive (GUI based), Time Sharing, Real Time and Distributed, Commonly used Operating System: UNIX, LINUX, Windows, Solaris, BOSS (Bharat Operating System Solutions); Mobile OS – Android, Symbian, IOS.

Utility Software: Anti-Virus, File Management tools, Compression tools and Disk Management tools (Disk Cleanup, Disk Defragmenter, Backup).

Open Source Concepts: Open Source Software, Freeware, Shareware, and Proprietary Software.

Application Software: Office Tools – Word Processor, Presentation Tool, Spreadsheet Package, Database Management System; Domain Specific tools – School Management System, Inventory Management System, Payroll System, Financial Accounting, Hotel Management, Reservation System and Weather Forecasting System.

Number System: Binary, Octal, Decimal, Hexadecimal and conversion between different number systems.

Internal Storage encoding of Characters: ASCII, ISCII (Indian Scripts Standard Code for Information Interchange), and UNICODE (for multilingual computing)

Microprocessor: Basic concepts, Clock speed (MHz, GHz), 16 bit, 32 bit, 64 bit, 128 bit processors; Types – CISC Processors (Complex Instruction Set Computing), RISC Processors (Reduced Instruction Set Computing), and EPIC (Explicitly Parallel Instruction Computing).

Memory Concepts: Units: Byte, Kilo Byte, Mega Byte, Giga Byte, Tera Byte, Peta Byte, Exa Byte, Zetta Byte, Yotta Byte.

Primary Memory: Cache, RAM, ROM

Secondary Memory: Fixed and Removable storage – Hard Disk Drive, CD/DVD Drive, Pen Drive, Blue Ray Disk.

Input Output Ports/ Connections: Serial, Parallel and Universal Serial Bus, PS-2 port, Infrared port, Bluetooth, Fire wire.

CHAPTER 1

1. Explain briefly the functions of ALU. 2

A) The Arithmetic and Logic Unit performs all the four arithmetical (+,-,*,/) and some logical (<,<=,>,>=,< >) operations. When two numbers are required to be added, these numbers are sent from memory to ALU where addition takes place and the result is put back in the memory. In the same way other arithmetic operations are performed.

For logical operations also, the numbers to be compared are sent from memory to ALU where the comparison takes place and the result is returned to the memory. The result of a logical operation is either TRUE or FALSE. These operations provide the capability of decision-making to the computer.

2. What are the basic operations of a computer.

A. Basic Operations of a Computer:

(i) Taking input from the input device and stores it in memory.

(ii) Processing according to the programs(instructions) and covert input to the output. This process will be performed in Microprocessor which consists Arithmetic and Logic Unit and Control unit.

(iii) Displaying output on any output device.

3. Write two types of Micro Computers? 1

A. Types of Micro Computers:

i) PDA (Personal Digital Assistants).

ii) Desktop Computers and Laptop (Notebook) Computers.

iii) Workstations.

4. Mention the types of computers depending on size and performance. 2

A. Embedded Computers, Micro Computers (PC, Laptop, tablets, etc), Mini Computers, Mainframe computers and Super computers.

5. How many bytes are in a terabyte? 1

A) 1024 X 1024 X 1024 X 1024 Bytes

(or) 2^{10} X 2^{10} X 2^{10} X 2^{10} Bytes (or) 2^{40} Bytes

6. How many bytes make 1 terabyte? 1

7. What is hardware? Give one example. 1

8. Differentiate between impact and non impact printers. 2

9. What is the electronic component used in second generation of computers. 1

10. Expand the following:

i)CPU ii) ROM iii)MICR iv)CD-R

A) (i) CPU - Central Processing Unit

(ii)ROM - Read Only Memory

(iii)MICR – Magnetic Ink Character Reader

(iv) CD –R – Compact Dist Recordable

11.Expand UPS. 1

A) Uninterrupted Power Supply

12) Write full form. 3

1) ROM 2) ATM 3) PDF 4) BIOS 5) UPS 6) DVD

Read Only Memory, Automatic Teller Machine, Portable Document Format, Basic Input Output Service/System, Uninterrupted Power Supply, Digital Versatile Disk.

13. Explain first generation of computer. 2

14. Which of the following is related to Ist generation computers? (A) Vacuum Tubes

15. Full form of ALU is ___. (Arithmetic Logic Unit)

16. 4 bits constitute to form ___ (A nibble)

17. Who is known as 'Father of Modern Computers'? (A) Charles Babbage

18. UNIVAC was the first commercial computer designed in USA. What is the full form of UNIVAC?

(A) Universal Automatic Computer

19. Who invented 'Analytical and Difference Engine'? (A) Charles Babbage

20. Name the two super computers developed in India? (A) PARAM,ANURAG

21. Define Hybrid computer? [1]

22. Explain booting process and its types. [2]

A) Booting Process is of two types – Warm and Cold

Cold Booting: When the system starts from initial state i.e. it is switched on, we call it cold booting or Hard Booting. When the user presses the Power button, the instructions are read from the ROM to initiate the booting process.

Warm Booting: When the system restarts or when Reset button is pressed, we call it Warm Booting or Soft Booting. The system does not start from initial state and so all diagnostic tests need not be carried out in this case. There are chances of data loss and system damage as the data might not have been stored properly.

CHAPTER 2

1. Define Software and its types.

A. Software: Software represents the set of programs that govern the operation of a computer System and make the hardware run.

Eg: Dos, MS-Word, C++,Visual Basic,Paint Brush, etc.

Software can be classified broadly into two categories:

(i)System Software: a)Operating System b) Language Processors

(ii) Application Software

2. Give an example of a Utility. 1

3. What is application software? Give example. 1

4.Write the names of two popular system softwares. 1

A) Operating Systems: DOS, Windows XP, Unix, Solaries.

Language Processors: Assembler, Compiler, Interpreter.

5. What do you understand by booting? Write its types. 2

6.Define operating system with its two functions. 2

7. What is an operating system? Give one example. Mention the types of Operating System. 3

8. What do you understand by compiler and interpreter? 2

9. Differentiate between freeware and shareware. 2

10. Explain SJN and FCFS scheduling. 2

11. Write different between application software and system software. [2]

12. Explain functional components of a Computer? [2]

A) Functional components of a computer: Input Unit, CPU, Main Memory and Output Unit.

An input unit takes the input and converts it into binary form so that it can be understood by the computer.

CPU (Central Processing Unit) has two components ie ALU performs arithmetic and logical operations and control unit which controls and guides the data flow, performs program execution, etc. There are two types of memories ie primary memory (RAM, ROM and Cache) and secondary memory.

The Output unit takes the output from CPU and converts output from binary form to human understandable form.

(1 mark for giving names of functional units(input/output/memory)

1 mark for explanation of the working of functional units.)

13. What function of operating system plays to manage memory. [1]

A) Memory management system

CHAPTER 3

1. Why do Computers use Binary Number System? (3)

A. In our real life we use decimal number system, which consists totally 10 digits. It is very difficult to design electronic equipment so that it can work with 10 different voltage levels On the other hand, it is very easy to design simple, accurate electronic circuits that operate with only two voltage levels. For this reason, almost every digital system uses the binary number system (base 2) as the basic number system of its operations, although other systems are often used in conjunction with binary.

In the binary system there are only two symbols or possible digit values, 0 and 1.

Even so, this base-2 system can be used to represent any quantity that can be represented in decimal or other number systems. So Computers use Binary Number System.

2.Base of Octal and Hexadecimal is ___ and ___. (1)

A. Base of Octal = 8

Base of Hexadecimal = 16

3.What is the weight or radix of Hexadecimal number system?1

4. Convert $(25)_{10}$ to binary

5. Convert $(AC.20)_{16} = (?)_2 = (?)_8$

A) $(AC.20)_{16}$

Binary equivalent of A = 1010
Binary equivalent of C = 1100
Binary equivalent of 2 = 0010
Binary equivalent of 0 = 0000
So, $(AC.20)_{16} = (10101100.00100000)_2$

10 | 101 | 100 | . 001 | 000 | 00

$(10101100.00100000)_2$ = 2 5 4 1 0 0

$= (254.100)_8$

6. Convert $(4A56)_{16} = (?)_{10} = (?)_8$

A) $(4A56)_{16} = 4 \times 16^3 + 10 \times 16^2 + 5 \times 16^1 + 6 \times 16^0$

$= 4 \times 4096 + 10 \times 256 + 5 \times 16 + 6 \times 1$

$= 16384 + 2560 + 80 + 6$

$= (19030)_{10}$

$(4A56)_{16} = (0100\ 1010\ 0101\ 0110)_2$

0 | 100 | 101 | 001 | 010 | 110

$= 4\ 5\ 1\ 2\ 6 = (45126)_8$

7. Find the 1's and 2's complement of 128. 2

A) Binary Equivalent of 128 = 10000000

1's Complement of 128 = 01111111

+1

2's Complement of 128 = 10000000

8. Convert according to given their base address (2)

$(4502)_{10} \rightarrow (\ )_{16} \rightarrow (\ )_2 \rightarrow (\ )_8$

9 a) Find the one's compliment of -13 in 8-bit form 1

b) Convert 10111101000111 into octal. 1

10. Represent -12 in one's complement method. (Each word consists 8 bits) 2

11. Expand ASCII, ISCII. 1

12. Write the full forms: (a) ENIAC (b) ASCII 1

A.ASCII – American Standard Code for Information Interchange.

ISCII - Indian Standard Code for Information Interchange.

13) What is the base of Decimal, Binary, Octal and Hexadecimal number systems?

(A) Decimal=10,Binary=2,

Octal=8 and Hexadecimal=16

14(A)Convert the following into its equivalent codes. [4]

(i) $(84)10 = (?)_2$ (ii) $(2C9)16 = (?)_{10}$

(iii) $(101010)2 = (?)_{10}$ (iv) $(3674)8 = (?)_2$

[Ans] (i) $(84)_{10} = (1010100)_2$

(ii) $(2C9)_{16} = (703)_{10}$

(iii) $(101010)_2 = (42)_{10}$

(iv) $(3674)_8 = (011110111100)_2$

15. Express -4 in 1's complement form. [1]

A) – 4 = 11111011 (+4=00000100)

CHAPTER 4

1. Differentiate between primary memory and secondary memory. Give examples of each type of memory. 2

A) The memory inside the CPU is primary memory (main memory) and the memory outside it is known as secondary (auxiliary) memory.

Primary Memory: RAM (Random Access Memory) and ROM (Read only Memory) comes under primary memory. RAM is volatile memory and ROM is non volatile memory. All the data and programs must be stored in RAM for execution. But the content of RAM is not permanent.

Eg: RAM, ROM.

Secondary Memory: Since primary memory has a limited storage capacity and is not permanent, secondary storage devices are used to store large amount of data permanently. There are various types of secondary devices available these days. Eg: Hard disks, Floppy disks ----- Magnetic Media CD ROMS, DVDs ----- Optical Media

2.Define the following: 2

i)DRAM ii) Serial Port

A) **(i) DRAM:** Dynamic Random Access Memory. DRAM consists of a transistor and a capacitor that's capable of storing an electric charge. Depending on the switching action of the transistor, the capacitor either contains no charge(0) or does hold a charge(1). DRAM is volatile ie contents are lost in the event of power failure. Regenerator circuits are there to refresh the memory.

(ii) Serial Port (COM or RS232C Ports): The serial port transfers data serially a bit at a time. Serial ports come in the form of 9 pin or 25 pin male connector.

3. Differentiate between RAM and ROM. 1

4. What is cache memory? Write its advantages. 2

5. Give any 2 examples of secondary memory devices. 1

6. Write short note on Hard Disk.. 2

A. Hard disk is a secondary memory device. The hard disk memories store information on one or more circular platters(disks) which are continually spinning. These rotating disks are coated with a magnetic material and stacked with space between them. Information is recorded on the surface of rotating disks by magnetic heads as tiny magnetic spots. These heads are mounted on access arms. Information is recorded in bands. Each bank of information on a given disk is called a track. The tracks are commonly divided into pie-shaped sections called sectors. In most systems, the minimum quantity of information which can be transferred is a sector. A motor rotates the disk at a rapid speed. Data are recorded on the tracks of a spinning disk surface and read from the surface by one or more read/write heads. The hard disks of today have storage capacity measured in giga bytes, most common being 160 and 250 GB.

(Concentric circles on the magnetized surface of the magnetic disks known as Tracks. The tracks on the disk surface are divided into invisible segments known as Sectors.)

7.Explain tracks & sectors in a hard disk. 2

8. Discuss the basic composition of a simple microprocessor. 2

9. Define compiler, Non-volatile memory, application software, USB port. 4

10. Which of the following are hardware and software? 2

(i) Capacitor (ii)Intenet Explorer (iii)Hard disk (iv)UNIX

A) (i) Capacitor - Hardware (ii) Internet Explorer – Software

(iii) Hard Disk – Hardware (iv) UNIX - Software

11) Differentiate between CISC and RISC processors. [2]

A) CISC (Complex Instruction Set Computing):

1. A large and varied instruction set.
2. Performs basic as well as complex functions.
3. All HLL support is done in Hardware.
4. Memory to memory addressing mode

RISC (Reduced Instruction Set Computing):

1. RISC system has reduced number of instructions.
2. Performs only basic functions.
3. All HLL support is done in software.
4. All operations are register to register.

12.Write two types of cache memory. [2]

Ans) L1 cache: It is small and is built inside the CPU. It is fast as compared to L2 Cache.

L2 cache: It is large but slower and is mounted on the motherboard.

XI – CHAPTERWISE OLD QUESTIONS

FOR MODELS PURPOSE ONLY NOT COMPLETE MATERIAL

C++ (UPTO FLOW OF CONTROL)

SYLLABUS

Unit-3: Introduction to C++

Getting Started: C++ character set, C++ Tokens (Identifiers, Keywords, Constants, Operators,), Structure of a C++ Program (include files, main function), Header files – iostream.h, iomanip.h, **cout, cin;** use of I/O operators (<<and>>), Use of endl and setw ( ), Cascading of I/O operators, compilation , Error Messages; Use of editor, basic commands of editor, compilation, linking and execution.

Data Types, Variables and Constants: Concept of Data types; Built-in Data types: **char, int , float** and **double;** Constants: Integer Constants, Character constants (- \n, \t, \b), Floating Point Constants, String Constants; Access modifier: **const;** Variables of built-in-data types, Declaration/Initialization of variables, Assignment statement, Type modifier: **signed, unsigned, long**

Operator and Expressions: Operators: Arithmetic operators (-,+,*,/,%), Assignment operator (=), c++ shorthands (+=, -=,*=,/=,%=) Unary operators (-), Increment (++) and Decrement (--) Operators, Relational operator (>,>=,<=,=,!=), Logical operators (!,&&,||), Conditional operator: <condition>?<if—true>:<if false>; Precedence of Operators; Automatic type conversion in expressions, Type casting;

UNIT 4: PROGRAMMING IN C++ :

Flow of control

Conditional statements: **if else,** Nested **if, switch..case..default,** use of conditional operator, Nested **switch..case, break** statement (to be used in **switch..case only); Loops**: **while, do – while, for** and Nested loops.

Header file Categorization	Header File	Function
Standard input/output functions	stdio.h	gets ( ), puts ( )
Character Functions	ctype.h	isalnum ( ), isalpha ( ), isdigit ( ), islower ( ), isupper ( ), tolower ( ), toupper ( )
String Function	string.h	strcpy ( ), strcat ( ), strlen( ), strcmp ( ), strcmpi ( ), strrev ( ), strupr ( ), strlwr ( )
Mathematical Functions	math.h	fabs ( ), pow ( ), sqrt ( ), sin ( ), cos ( ), abs ( )

Introduction to user-defined function and its requirements.
Defining a function; function prototype, Invoking/calling a function, passing arguments to function, specifying argument data types, default argument, constant argument, call by value, call by reference, returning values from a function, calling functions with arrays, scope rules of variables: local and global variables.
Relating to Parameters and return type concepts in built-in functions.

STRUCTURED DATA TYPE

ARRAYS: Introduction to Array and its advantages.
One Dimensional Array: Declaration/initialization of One-dimensional array, Accepting array elements, accessing array elements, manipulation of array elements (sum of elements, product of elements, average of elements, linear search, finding maximum/minimum value)
Declaration / Initialization of a String, string manipulations (counting vowels/ consonants/ digits/ special characters, case conversion, reversing a string, reversing each word of a string)
Two-dimensional Array: Declaration/initialization of a two-dimensional array, inputting array elements, accessing array elements, manipulation of Array elements (sum of row element, column elements, diagonal elements, finding maximum / minimum values)
User-defined Data Types: Introduction to user defined data types.

STRUCTURE: Defining a Structure (Keyword Structure), declaring structure variables, accessing structure elements, passing structure to functions as value and reference, argument/parameter,
function returning structure, array of structure, passing an array of structure as an argument/ a parameter to a function.
Defining a symbol name using **typedef** keyword and defining a macro using **#define preprocessor** directive.

THEORY QUESTIONS

(1) Who is the developer of C++ language?
(A) Bjarne Stroustrup

2) What is "Code Generation"?
Can a program be executed before it? [2]

A) A compiler translates the corrected program text into object or assembly instruction text understood by the computer. This process of translation is called code generation.
A program cannot be executed before code generation.

3.What do you meant by a lexical unit? Give an example. 1

A) Token: The smallest individual unit in a program is known as a token (Lexical Unit).
There are 5 types of tokens.
(i) Keywords (ii) Identifiers (iii) Literals
(iv) Punctuators (v) Operators

4. Explain tokens in details with example. 3

A. **Token:** The smallest individual unit in a program is known as a token (Lexical Unit).
There are 5 types of tokens.
(i) Keywords (ii) Identifiers (iii) Literals
(iv) Punctuators (v) Operators.

(i) Keywords: Keywords are the words that convey a special meaning to the language compiler. These are reserved for special purpose and must not be used as normal identifier names. Original C++ consists 48 keywords and ANSI C++ consists 63 keywords. Eg: int, switch, while, etc.

(ii)Identifiers: Identifiers are the names given variables, objects, classes, functions, arrays, etc.
Eg: rno, myfile, etc.

(iii)Literals (Constants):are data items that never change their value during a program run.

Different types of literals:
(i) Integer constant (ii) Floating Constant (iii) Character Constant (iv) String Constant. (v) boolean literal

(iv)Operators : The operations (specific tasks) are represented by operators and the objects of the operation(s) are referred to as operands. They can be classified into 3 categories depending on number of operands it is working:
(a) Unary operators: Works on single operand.
Eg: ++ , -- , sizeof, etc
(b) Binary operators: Works on two operands.
Eg: +, - , *, /, %, <, >, etc.
(c)Ternary operator: Works on three operands.
Eg: Conditional operators.
(exp1)?exp2:exp3

(v) Punctuators (Separators): [,], (,), { , } , ;, :, #, etc.

5. What is a token? Explain about tokens. 5

6. Write any four escape sequence characters? 1

A. (i) \a - Audible bell (ii)\b - Backspace
(iii)\n - Newline (iv) \r – Carriage return
(v) \t - Horizontal tab (vi) \0 – Null

7. Classify the following variable names of C++ into valid and invalid category [2]

(i) 1no (ii) num 1 (iii) num
(iv) num1num (v) num+1 (vi) num.1

A) Valid : num, num1num
Invalid: 1no, num 1, num+1, num.1

8. Define operators and its types? 4

A. **Operators :**The operations (specific tasks) are represented by operators and the objects of the operation(s) are referred to as operands. They can be classified into 3 categories depending on number of operands it is working:

(a) Unary operators: Works on single operand.
Eg(i) ++ (Increment operator). Used to increment 'one' to the existing value. A=10; A++;
After the execution of above statement, A value will becomes as 11.
(ii) - - (Decrement Operator). Used to decrement 'one' to the existing value.
(iii) sizeof( ): This is used to return the number of bytes for the given data type or variable.

```
int a;
sizeof(a);          //returns 2 since a is of type int.
```

(iv) Unary + : It will give the value with same sign.
A=10; +A will gives 10 only.
A= -10; +A will gives -10.

(v) Unary - : It will give the value with opposite sign.
A=10; -A will gives – 10.
A=-10; -A will gives 10.
(vi) !: Not
(b) Binary operators: Works on two operands.
Eg: Arithmetic Operators: + , - , * , / , %
+ for addition on two operands. - for subtraction
* for multiplication / for quotient
% for remainder.
Relational Operators: < ,<=, >, >=, !=, = =.
Logical Operators: &&, || Etc.,
(c)Ternary operator: Works on three operands.
Eg: Conditional operators. (exp1)?exp2:exp3
Here when the expression 1 becomes true expression 2 will be executed and if it becomes false, then expression 3 will be executed.

9. Explain various types of Constants with example. 3
A. Constants (Literals): are data items that never change their value during a program run.
Different types of literals…
(i) Integer constant (ii) Floating Constant
(iii) Character Constant (iv) String Constant.
(i) Integer Constant: Integer constants are whole numbers without any fractional part.
Three types of integer constants.
(a)Decimal integer constants (b) Octal Integer Constants
(c) Hexadecimal integer Constants
Eg: 789 – Decimal integer constant
032 – Octal integer constant
0xFA2 – Hexadecimal integer constant.
(ii) Floating Constants (Real Constants): Real constants are numbers having fractional Form. These may written in fractional form or exponential form.
Eg: 17.45 - Fractional form
0.1745E2 – Exponential form
(iii)Character Constants: It is a single character kept in between pair of ' '. Any ASCII Character can be represented as a character constant. An escape sequence, which consists non-graphic characters can be represented through a character constant.
Eg: 'n', '\n', 'z','\a', '\t', 'k', etc.
(iv) String Constants: Multiple character constants are treated as string literals. A string Literal is a sequence of characters surrounded by double quotes. Each string literal is by default added with a special character '\0'.
Eg: "abc" - size is 4. "Navodaya" – size is 9.

10. How many keywords are there in C++ and in ANSI C++? 1
A. 48 (in C++) and 63 (in ANSI C++)
(As per C++11 standard, 84 keywords)

11. What is a Source Code? 1
A. The program written in a high level language (here in C++) is called as source code. (The program that is converted into the machine language is called as object code).

12. What are fundamental data types? Write the keywords for them. 2
A) Fundamental (atomic) data types are those that are not composed of other data types.
There are 5 fundamental data types: (i) int (ii) float (iii) double (iv) char (v) void.

13. How many bytes take to store a variable of type 'int'? What is the range of int? 1
A. 2 Bytes. (Range : -32768 to 32767)
(Others : Char – 1 byte, float – 4 bytes, long – 4 bytes, double – 8 bytes, long double – 10 bytes, long long – 8 bytes)

14. Explain postfix and prefix operators in C++ with suitable example. 4
A) (i) ++ (Increment Operator): Used to increment one value to the existing value.
If a=10; a++;
Then a value will becomes 11.
(ii) - - (Decrement Operator): Used to decrement one value to the existing value.
If a=10; a--; Then a value will becomes 9..
Note: While we use ++ and - - operators, there are two notations.
(A) **Prefix :** If operator comes before operand, it is called as prefix notation.
Eg: ++a; - - a;
Prefix notation means change and store.
a=10; b= ++a; //Now a=11,b=11.
a=10; b= - - a; //Now a=9,b=9.
(B) **Postfix:** If operator comes after operand, it is called as postfix notation.
Eg: a++; a- -;
Postfix notation means store and change.
a=10; b= a++; //Now a=11,b=10.
a=10; b= a- -; //Now a=9,b=10.

15. What is type conversion? Explain the types of type conversion. 2
A) Type Conversion: The process of converting one predefined type into another is called type conversion. These are of two types.
(i) Implicit type conversion (Type promotion) : An implicit type conversion is a conversion performed by the compiler without programmer's intervention. In this type conversion, C++ compiler converts all operands up to the type of the largest operand.
Eg: int a=20;
float b=33.77;
cout<<a+b; // Then it displays 53.77 as float is the biggest type.
(ii) Explicit type conversion (Type casting) : An explicit type conversion is user
defined that makes an expression to be of specific type. Explicit type conversion will be done by the programmer.
Syntax: (type) expression;
Eg: float a=77.37;
cout<<(int)a; // Will makes a as an int. Now 77 will be displayed.

16. What is a variable? 1
A. Variable: A variable is a named memory location. The value of a variable varies.
The name of a variable should be a valid identifier.

17 Explain b) switch statement 2
A. Switch: Switch is a selective statement.
C++ provides a multiple branch selection statement known as switch. This selection statement successively tests the value of an expression against a list of integer or character constants. When a match is found, the statements associated with that constant are executed.
Syntax:

```
switch(expression)
{ case constant1: statement sequence 1;
                  break;
  case constant2: statement sequence 2;
                  break;
  case constant3: statement sequence 3;
                  break;
  - - - - - - - - - -
  - - - - - - - - - -
  case constant-n: statement sequence n;
                  break;
  [default: statement sequence];
}
```

18. What are Iteration Statements? Explain them in detail.6

A. The iteration statements allow a set of instructions to be performed repeatedly until a certain condition is fulfilled. The iteration statements are also called loops or looping statements. C++ Provides three kings of loops. (i) for (ii) while (iii)do-while loop. All three loop constructs of C++ repeat a set of statements as long as a specified condition remains true. This specified condition is generally referred to as a loop control. For all three loop statements, a true condition is any nonzero value. A zero value indicates a false condition.

Parts of a loop:

(i) Initialization expression(s) (ii)Test expression.
(iii)Update expression(s) (iv)Body of the loop.

(i) For loop:

Syntax: for (initialization expression(s); test expression;
update expression(s))
body of the loop;

Firstly, initialization expression is executed. Then the test expression is evaluated, which results into either true(1) or false(0). If the test expression is true, then the body of the loop executed otherwise control come out of the for loop. After execution of the body of the loop, then the update expression will takes place. Then condition will be tested….the process will repeat until the condition evaluates to false.

Eg: for (i=1;i<=10;i++)
cout<<i;

(ii) While loop (Entry controlled loop):

Syntax: while (test expression)
body of the loop;

The loop variable must be initialized before enter into the while loop and it should be updated inside the boy of the while loop.

The condition will be tested before entering into the loop. So if initially the condition evaluates to false, then the loop will not execute at least once.

Eg: i=1;
while (i<=10)
{ cout<<i;
i++;
}

(ii) Do- While loop (Exit controlled loop):

Syntax: do
{body of the loop;

} while (test expression)

The condition will be tested after execution of the loop for once. So if initially the condition evaluates to false, then the loop will execute at least once.(since the check will be performed at the end)

eg: i=1;
do
{ cout<<i;
i++;
}while (i<=10);

19. Explain about 'for' loop. 2

DIFFERENCES

1. Differentiate between 'a' and '\a'. 1

2. What is the difference between the 'a' and "a"? 1

A. 'a' is a character literal so it takes 1 Byte to represent.
"a" is a string literal so it takes 2 Bytes to represent.

3. Differentiate between / and %. 1

A. / will gives quotient and % will gives reminder if both numerator and denominator are integers.

/ will work as arithmetic division if any one or both numerator and denominator are real numbers.

Ex : 25 / 3 will gives result 8 (Quotient)
25 % 3 will gives result 1 (Reminder)
25.0 / 3 will gives result 8.33 (Real number).

4. What is the difference between = and = = ? 1

5.What is the difference between '=' and '=='? 1

A. = is assignment operator. This operator stores the right side value into left side variable. = = is relational operator. It will check whether right side and left side operands are equal or not and returns a true(1) when both are equal, false(0) otherwise.

6.Differentiate between Break & Continue.

Break: A break statement skips the rest of the loop and jumps over to the statement following the loop. **(**The break statement enables a program to skip over part of the code. A break statement terminates the smallest enclosing while, do-while, for or switch statement. Execution resumes at the statement immediately following the body of the terminated statement. **)**

Continue: The continue statement skips the rest of the loop statement and causes the next iteration of the loop.

7. Distinguish between if and switch statement.

A. The if-else and switch both are selection statements and they both let you select an alternative out of given many alternatives by testing an expression. But there are some differences in their operations.

(i) The switch statement differs from the if statement in that switch can only test for equality whereas if can evaluate a relational or logical expression. Ie multiple conditions.

(ii) The switch statement selects its branches by testing the value of same variable whereas the if else construction lets you use a series of expressions that may involve unrelated variables and complex expressions.

(iii) The if-else can handle ranges whereas switch cannot. Each switch case label Must be a single value.

(iv) The if-else statement can handle floating point tests also where as switch cannot handle floating point tests.

(v) The switch case label value must be a constant. So, if two or more variables are to be compared, use if-else.

8. Differentiate between while and do-while loop. 1

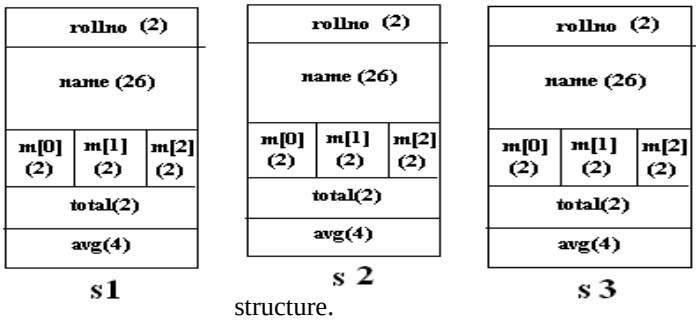
A) Differences between while & do-while:

While: It is an entry controlled loop. In while, the test-expression is evaluated at the beginning of the loop. If the test expression evaluates to false for the first time itself, the loop is never executed.

Do-while: It is an exit controlled loop. In Do-while, the test-expression is evaluated at the end of the loop. Even if the test expression is initially false, the loop will execute for at least once.

9. Differentiate between unions and structures with help of an example. 2

A)<u>Structure:</u> A structure is a collection of elements (variables) of different data types. The keyword struct is used to construct a structure.



Syntax: structstructtag
{ datatype variable;
datatype variable.

}structure variables;

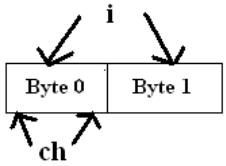
Eg: struct student
{ introllno;
char name[26];
int m[3];
int total;
float avg;
}s1,s2;
struct s3;

Referencing Structure variables:
Structurevariable.elementname;

Eg: s1.rollno;
s2.name;
s3.marks[2];

Union: A union is a memory location that is shared by two or more different variables (generally shared by two or more different times).

Eg: union share
{ int i;
Char ch;
};
union share us;
us.i = 20;
cout<<us.ch;



At any point, you can refer to the data stored in a us as either an integer or a character.

Difference between a union and a structure:
A union is a memory location that is shared by two or more different variables.
A structure have separate memory location for every variable. In the above example for structure variable S1, 40 Bytes will be reserved and For union variable us, 2 bytes will be reserved.

10.The break statement causes an exit from …… [1]
A) The break statement causes an exit from the smallest Enclosing while, do-while, for or switch statement.
(Or) The break statement causes an exit from the innermost Loop or switch.

11) The exit statement causes an exit from …….. [1]
A) The exit statement causes an exit from the program it appears in.

EQUIVALENT STATEMENTS

1. Write the equivalent C++ statement for the following: 1
x^5+ax^3
A. pow(x,5) + a*pow(x,3)

2.Convert the following equations to C++ statements. 2
i) $s = 1+1/x+1/x^2+1/x^3$
A) s = 1 + 1/x + 1/(x*x) + 1/(x*x*x);
ii)$V = 4/3\prod r^3$
A) V = 4/(3*3.1415*r*r*r);

3. Write the equivalent C++ expression for the following arithmetic expressions: 2
i)$(a+b+c)^3+2ab+ab^2$
A) pow((a+b+c),3)+2*a*b+a*b*b
ii)$Ut + ½ at^2$ A) U*t+(1/2)*a*t*t
iii)$\sqrt{(p+q)+x^2}$ A) sqrt(p+q+x*x)
iv)$P(1+r/100)^n$ A) p*pow((1 + (r /100),n)

4. Write the equivalent C++ expressions- [2]
(i) p=2(l+b) (ii) z=2(p/q)2
(iii) s=1/2mv2 (iv) x=-b+√(b2-4ac)/2a

Ans) Equivalent expressions are :
a) p=2*(l+b);
b) z=2*pow((p/q),2)) or 2*p/q*p/q;
c) s=1/2*m*v*v; or s=1/2*m*pow(v,2);
d) x=-b+sqrt(b*b-4*a*c)/2*a;
or x=-b+sqrt(pow(b,2)-4*a*c)/2*a;

5.Write an equivalent while loop for the following for loop:2
for (int i=2, sum=0 ; i<=20 ; i=i+2)
sum += i ;

Ans)
int i=2;
sum=0;
while(i<=20)
{ sum+=i;
i=i+2;
}

6.Rewrite the following code fragment using switch: [2]
if (ch == 'N')
cout<<"NAVODAYA" ;
if (ch == 'V')
cout<<"VIDYALAYA" ;
if (ch == 'S')
cout<<"SAMITI" ;
else
cout<<"INVALID INPUT ";

Ans)
switch(ch)
{ case 'N': cout<<"NAVODAYA";break;
case 'V' :cout<<"VIDYALAYA";break;
case 'S': cout<<"SAMITI";break;
default: cout<<"INVALID INPUT";
}

7.Write one example of infinite loop using any one looping construct. [1]

C++ CODE

1. In each of the following cases show how the comment can be placed in a program: 2
i)Add a comment sum of three numbers to the statement:
sum=a+b+c;
A) sum = a+b+c; //sum of three numbers
ii)Add the comment End of Function with the '}'.
A) } //End of Function

2) What will the size of following ? 2
a) "jnv" b)'\a' c)3.13 d)"navodaya"
A) a) 4 bytes b) 1 byte c) 4 bytes d) 9 bytes

3.a) What will the size of constants: 2
(i)'\0' (ii)"\a" (iii)7.88 (iv)39872
A) 1 byte ii)2 bytes (iii) 4 bytes (iv) 2 bytes (if it is represented using unsigned int) (or) 4 bytes (if it is long)

4. What will be the result of the following two expressions if i=15 initially? 2
a) i++ >= 15 A) True
b) ++i <= 15 A) False

5. What will be the result of the following two expressions if i=10 initially? 2
1) ++i<=10 A) False
2)i++<=10 A) True

6. void main( ) 2
{ int val, res, n=1000;
cin>>val;
res=n+val>1750 ?400:200;
cout<<res;
}
1. If val=750; the res=? A) 200
2. Name of the operator used in the program?
A) Conditional Operator

7. Convert the following if-else to a single conditional statement. 1
if(qty>=20)
order=max+5;
else
order=max;
A) order = (qty>=20)?max+5:max;

FUNCTIONS

1. Write the name of the header files to which the following belong **2**

(i) clrscr ( ) A) conio.h

(ii) exit( ) A) process.h

2. Name the header file to which the following functions belongs? **1**

(a) pow( ) A) math.h

(b) clrscr( ) A) conio.h

3.Write the names of the header files of the following functions.

i)getch() ii) isalpha() iii)strcpy() iv)sqrt()

A) (i) getch( ) - conio.h (ii)isalpha( ) – ctype.h

(iii)strcpy( ) - string.h (iv)sqrt( ) - math.h

REWRITE

1) Rewrite the following C++ code after removing any/all syntactical errors with each correction underlined. 3

Note : Assume all required header files are already being included in the program.

```
void main
{ clrscr( );
  int a,b,c;
  for(i=0;i<20;i++)
  {  cout<<"enter the value of a,b,c";
     cin>>a>>b>>c;
     S=a+b+c
     cout<<s;
   }
    getch;
 }
```

Ans:

```
void main( )
{ clrscr( );
  int a,b,c;
  int S;
  for(i=0;i<20;i++)
  {  cout<<"enter the value of a,b,c";
     cin>>a>>b>>c;
     S=a+b+c;
     cout<<S;
   }
    getch( );
 }
```

2. Rewrite the following code after removing the syntactical errors. **2**

```
    Void main(  )
    [     Int a,b
          a=10;
          b=20;
          c=a+b,
          cout<<"The addition of two numbers = "<<'c';
    )
```

Ans)

```
    void main(  )
    {     int a,b,c;
          a=10;
          b=20;
          c=a+b;
          cout<<"The addition of two numbers = "<<c;
    }
```

3. Find errors if any in the following statements. **2**

```
cout<<"X=";X ;
cin>>x;>>y;
cout<<"Enter a number";
cin>>x;
```

A. Errors in Line 1:

(a) ; is given in place of <<

(b) X is given in place of x

Error in Line 2:

; is given in the middle of the line cin>>x;>>y;

Corrected version:

```
cout<<"X="<<x;
cin>>x>>y;
cout<<"Enter a number";
cin>>x;
```

4) Identify the errors in the following code segment and also write the corrected program. **[2]**

```
#include <iostream.h>
int main( )
{
int number, class , sum;
cout<<"Enter a number and class:" ;
cin >>number >> class ;
number + class = sum ;
cout<<"Sum ="<<sum ;
}
```

Ans)

```
#include <iostream.h>
void main( )
{
int number, Class , sum;
cout<<"Enter a number and class:" ;
cin >>number >> Class ;
sum=number + Class;
cout<<"Sum ="<<sum ;
}
```

OUTPUT

1) Write the output of the following code: **3**

```
#include<iostream.h>
int main( )
{  unsigned int a,b,c,l;
       a=1;
       b=2;
   cout<<a<<b;
   for(i=0;i<5;i++)
   {   c=a+b;
       cout<<c;
       a=b;
       b=c;
   }
}
```

A) 123581321

2.What will be the output of the following program? **2**

```
int main()
{ int i=0,x=0;
  for(i=1;i<10;i*=2)
  {  x++;
     cout<<x;
   }
  cout<<"\n"<<x;
}
```

A) 1234
4

3. What will be the output produced by following code fragment: **[1]**

```
int i , j;
for( i=10 ; i<=50 ; i+=10)
        j= i /2;
cout<< j<< " " ;
```

A) Output : **25**

4. What will be the output of the following program segment? **[2]**

If input is as: (a) g (b) b (c) e (d) p

```
cin >>ch;
switch (ch)
{ case 'g': cout<<"Good";
```

```
  case 'b': cout<<"Bad"; break;
  case 'e': cout<<" Excellent"; break;
  default: cout<<" wrong choice";
}
```

A)

Input	Output
g	GoodBad
b	Bad
e	Excellent
P	wrong choice

5. What output will be the following code fragment produce? [3]

```
void main( )
{  int val, res, n=1000;
   cin>>val;
   res = n+val >1750 ? 400:200;
   cout<<res;
}
```

(i) if val=2000 A) 400
(ii) if val=1000 A) 400
(iii) if val=500 A) 200

OUTPUTS USING PREFIX & POSTFIX

1.What will be the result of the following two expressions if i=10 initially? 2

i) `cout<<i<<i++<<++i;`
A) 121111
ii) `cout<<i<<i+1<<++i;`
A) 111211

2. Find the output of the following: 3

```
void main( )
{  int a=7, b=8,c ;
   c= a++ + ++a;
   cout<<c;
   c=- - a + ++a + a++;
   cout<<c;
}
```

A)

3. What will be value of x for following expression 2

if a=5, b=7, c=10, d=13.
(i) x=c!=a*b?++b: (b=16,b+=2, b - -)
(ii) if a=10, x=a++%5+ ++a + ++a
A)

4. Predict the output of the following code segment: [2]

```
int n=7 ;
cout<<" - -n= "<< - -n <<"\n";
cout<<"n= "<< n- - <<"\n" ;
```

A) output :
--n = 6
n = 6

C++ PROGRAMS

1. Write a program to find out simple interest. 2

A)

```
#include<iostream.h>
#include<conio.h>
void main( )
{  float p,t,r,i;
   clrscr( );
   cout<<"\nEnter the value of principal, time and
         rate of interest: ";
   cin>>p>>t>>r;
   i=(p*t*r)/100;
   cout<<"\nSimple Interest = "<<i;
   getch( );
}
```

2. Write a program to input three integers and print the largest of three. 3

```
#include<iostream.h>
#include<conio.h>
void main( )
{  int a,b,c;
   clrscr( );
   cout<<"Enter any three numbers: ";
   cin>>a>>b>>c;
   if(a>b)
   {   if(a>c)
          cout<<"\n The largest number = "<<a;
       else
          cout<<"\n The largest number = "<<c;

   }
   else
   {   if(b>c)
          cout<<"\n The largest number = "<<b;
       else
          cout<<"\nThe largest number ="<<c;
   }
   getch( );
}
```

3. Write a program to swap the two numbers without using third variable. 3

```
#include<iostream.h>
#include<conio.h>
void main( )
{    int a,b;
     clrscr( );
     cout<<"\nEnter any two numbers: ";
     cin>>a>>b;
     cout<<"\nThe values of a = "<<a<<" and b = "
           <<b<<" before swapping";
     a=a+b;
     b= a – b;
     a=a – b;
     cout<<"\nThe values of a = "<<a<<" and b = "
          <<b<<" after swapping";
     getch( );
}
```

/* Write a program to swap the two numbers using third variable. 3

```
#include<iostream.h>
#include<conio.h>
void main( )
{    int a,b,temp;
     clrscr( );
     cout<<"\nEnter any two numbers: ";
     cin>>a>>b;
     cout<<"\nThe values of a = "<<a<<" and b = "
          <<b<<" before swapping";
     temp=a;
     a=b;
     b=temp;
     cout<<"\nThe values of a = "<<a<<" and b = "
          <<b<<" after swapping";
     getch( );
}                    */
```

4.Write a program to convert the temperature reading in Fahrenheit to Celsius. 2

A)

```
#include<iostream.h>
#include<conio.h>
void main( )
{  doublec,f;
clrscr( );
cout<<"\nEnter the temperature in Fahrenheit: ";
cin>>f;
c=(5/9.0)*f-32;
```

```
    cout<<"\nThe equivalent temperature in Celcius: "<<c;
    getch( );
    }
```

5. Write a program to input any number and print the output in the following manner: 3

Ie Suppose input is 4 then output must be

```
Your number is      4
It's square is      16
And square root is  2
```

Ans)

```
#include<math.h>
#include<iostream.h>
void main( )
{  int n;
   cout<<"\nEnter any number: ";
   cin>>n;
   cout<<"\nYour number is  "<<n;
   cout<<"\nIt\'s square is "<<n*n;
   cout<<"\nAnd square root is "<<sqrt(n);
}
```

6. Write a program to find the roots of a quadratic equation. 3

(Where $d=b^2 - 4ac$,
Condition: (1) d=0, Roots are equal or distinct.
(2) d>0, Roots are real
(3) otherwise, Roots are complex ie imaginary.)

7. Write a program to find the roots of a quadratic equation. 2

A)

```
#include<iostream.h>
#include<conio.h>
#include<math.h>
void main( )
{    clrscr( );
     double d1,d2,b,a,c,d;
     cout<<"\nEnter the value of b,a and c: ";
     cin>>b>>a>>c;
       d=(b*b-sqrt(4*a*c));
     if(d==0)
     cout<<"\nRoots are equal or distinct";
     else if(d>=0)
     cout<<"\nRoots are Real";
     else
     cout<<"\nRoots are complex..ie Imaginary";
       d1=(-b+d)/(2*a);
       d2=(b+d)/(2*a);
     cout<<"\nD1: "<<d1;
     cout<<"\nD2: "<<d2;
     getch( );
 }
```

8. Write a program to display the following output, by asking N value from the user. 2

2,6,10,14,18,22,…….,N

Ans)

```
#include<iostream.h>
void main( )
{   int i,N;
    cout<<"\nEnter the value of N";
    for(i=2;i<=N;i=i+4)
        cout<<i<<",";
}
```

9. Write a do-while loop that displays numbers 1, 3, 5, 7… 17, 19 [2]

Ans)

```
int i=1;
do
{  cout<<i<<',';
   i=i+2;
} while(i<=19);
```

10 .Write a program to print the following series 4

1 -4 7 -10 ……. -40

Ans)

```
#include<iostream.h>
void main( )
{  int i,t=0;
   for(i=1;i<=40;i=i+3)
   {    t++;
        if (t%2==1)
             cout<<i<<'\t';
        else
             cout<<-i<<'\t';
   )
}
```

11. Write a program to check that the given number is prime number or not. 4

```
#include<iostream.h>
#include<conio.h>
void main( )
{ int i,mul,n;
  i=1; mul=0;
  clrscr( );
  cout<<"\n Enter any number: ";
  cin>>n;
  while(i<=n)
  {  if(n%i==0)
           mul++;
     i++;
  }
  if(mul==2)
    cout<<n<<" is a prime number";
  else
    cout<<n<<" is not a prime number";
  getch( );
}
```

12) Write a C++ program to print the Fibonacci series upto the N terms. [3]

i.e. 0 1 1 2 3 5 8 …… N

```
#include<iostream.h>
#include<conio.h>
void main()
{  int n,first=0, second=1,i,third;
  clrscr();
  cout<<"\nHow many numbers you want to
              generate the fibonacci numbers..";
  cin>>n;
  cout<<"Fibinacci series are :\n";
  cout<<endl<<first<<"\t"<<second<<"\t";
  for(i=3;i<=n;i++)
  {      third=first+second;
         cout<<third<<"\t";
         first=second;
         second=third;
   }
  getch();
}
```

13. Write a menu driven programme using switch statement and the output of the program work like this (4)

```
  MAIN MENU
 (1) SUM
 (2) SUBTRACTION
(3) MULTIPLICATION
(4) DIVISION
(5) WRONG CHOICE
ENTER ANY TWO NO: 5  6
ENTER YOUR CHOICE  3
MULTIPLICATION = 30
```

Ans)

```
#include<iostream.h>
#include<conio.h>
void main( )
{int choice;
  float a,b;
  clrscr( );
  cout<<"\NMAIN MENU";
  cout<<"\n(1) SUM";
  cout<<"\n(2) SUBTRACTION";
  cout<<"\n(3) MULTIPLICATION";
  cout<<"\n(4) DIVISION";
 cout<<"\NWRONG CHOICE";
 cout<<"\NENTER ANY TWO NO: ";
 cin>>a>>b;
 cout<<"\NENTER YOUR CHOICE: ";
cin>>choice;
switch(choice)
{ case 1: cout<<"\nADDITION = " <<a+b; break;
  case 2: cout<<"\nSUBTRACTION = " <<a+b; break;
 case 3: cout<<<<"\nMULTIPLICATION = " <<a+b;
          break;
case 4: cout<<<<"\nDIVISION = " <<a+b; break;
case 5: cout<<"\NWRONG CHOICE";  break;
default: cout<<"\nYOU HAVE TO ENTER PROPER
CHOICE";
 }//End of switch
 getch( );
}while(choice!=5);
}
```

14) Write a program to process 4 function calculator using switch and do-while.

```
#include<iostream.h>
#include<conio.h>
void main( )
{    int choice;
     float a,b;
     do
     {   clrscr( );
         cout<<"\n Enter any two numbers: ";
         cin>>a>>b;
         cout<<"\n1.Addition";
         cout<<"\n2. Subtraction";
         cout<<"\n3. Multiplication";
         cout<<"\n4. Division";
         cout<<"\n5. Exit";
         cout<<"\n\n\n Enter your choice: ";
         cin>>choice;
         switch(choice)
       { case 1: cout<<"\nThe addition of two numbers:
                        "<<a+b; break;
          case 2: cout<<"\nThe subtraction: "<<a-b; break;
          case 3: cout<<"\nThe multiplication of two
                           numbers: "<<a*b; break;
          case 4: cout<<"\nThe division: "<<a/b; break;
          case 5: exit(0);  break;
          default: cout<<"\nYou have to enter proper choice";
     }//End of switch
     getch( );
     }while(choice!=5);
}
```

15. Write a program to print the following series. 4

```
*
* *
* * *
* * * * upto n no of line.
```

Answer:

```
#include<iostream.h>
#include<conio.h>
void main( )
{  int i,j,n;
   cout<<"\nEnter the value of n: ";
   cin>>n;
   for(i=1;i<=n;i++)
   {    for(j=1;j<i;i++)
            cout<<' * ';
        cout<<endl;
    }
}
```

16) Write a program to print the following output: [3]

```
5 5 5 5 5
4 4 4 4
3 3 3
2 2
1
```

Answer)

```
#include<iostream.h>
#include<conio.h>
void main()
{ clrscr();
  int i,j;
  for(i=5;i>=1;i--)
  {    for(j=1;j<=i;j++)
       {    cout<<i;
       }
       cout<<"\n";
  }
  getch();
}
```

FUNCTIONS

1.Differentiate between call by value and call by reference with help of an example. 4

A)(i)In call by value, actual arguments will be copied into the formal parameters.

In call by reference, formal parameters are references to the actual arguments.

(ii)In call by value, if any modification is occurred to the formal parameter, that change will not reflect back to the actual argument.

In call by reference, if any modification is occurred to the formal parameter (reference to the actual argument), the actual argument value will be changed.

(iii) We should go for call by value when we don't want to modify the original value.

We should go for call by value when we want to modify the original value.

(iv) Example:

```
void Change(int a, int &b)
{        a= 2*a;
         b=20;
 }
```

Here a is called by "call by value" method and b is called by "call by reference" . So as the value of a is changed, actual argument for a will not be changed, as the value of b is changed, actual argument for b will be changed.

2. What is a function? Write a program to find the factorial value of any number using function. 3

3.Write a program to find the factorial of a number recursive function. 3

A)

```
#include<iostream.h>
#include<conio.h>
long f=1;
long factorial(int n)
{  if (n==0)
       return f;
   else
       f=n*factorial(n-1);
}
```

```
void main( )
{  clrscr( );
  long num;
  cout<<"\nEnter the number to which you want to find
          factorial: ";
  cin>>num;
  cout<<"\nThe factorial of the number = "
        <<factorial(num);
  getch( );
}
```

4. Write a program to find the factorial of 10 using recursion? 4

A.
```
#include<iostream.h>
#include<conio.h>
long f=1;
long factorial(int n)
{if (n==0)
    return f;
 else
    f=n*factorial(n-1);
}
void main( )
{ clrscr( );
  cout<<"\nThe factorial of the number 10 = "
        <<factorial(10);
  getch( );
}
```

5.Write a program to find the total number of characters, lines and words in a paragraph of text. 4

A)
```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
void main( )
{   char str[300];
   int i,charcount=0,words=1,lines=1;
   clrscr();
   cout<<"\nEnter the Paragraph ie message: \n";
   gets(str);
   for(i=0;str[i]!='\0';i++)
   {  charcount++;
     if(str[i]==' ')
          words++;
     if (charcount%80==0)
          lines++;
   }
   cout<<"\nNumber of Characters in the entered message: "
                          <<charcount;
   cout<<"\nNumber of Words in the entered messge: "<<words;
   cout<<"\nNumber of Lines in the entered message: "<<lines;
   getch( );
}
```

6.Explain any two string handling functions with syntax and examples. 2

A) String functions:

Function	Description	Example
1. strcat – string concatenation. char *strcat (char *str1, const char *str2)	This function concatenates a copy of str2 to str1 and terminates str1 with a null. str1 should be large enough to hold both its original contents and those of str2	char *str1=”One”; char *str2=”Two”; strcat(str2,str1); cout<<str2; (It will print “TwoOne”).
2. strcmp – string compare int strcmp (const char *str1, const char *str2)	This function alphabetically compares two strings and returns –ve value if str1 is less than str2, 0 if str1 is equal to str2, and >0 (+ve value) if str1 is greater than str2.	char *str1=”Ace”; char *str2=”Bag”; strcmp(str2,st1) returns +ve value (>0 value) but strcmp (str1,str2), return –ve value.
3. strcpy – string copy char *strcpy(char *str1, const char *str2)	Copies the contents of str2 into str1	char *str1=”Ace”; char *str2=”Bag”; strcpy(str1,str2); cout<<str1; (It will print “Big”);
4 strlen – string length int strlen(char *str)	Returns the length of the null terminated string pointed to by str. The null is not counted.	strlen(“Raju”) would give 4.

7. Write a program to compare two strings without using functions. (3)

A.
```
//Program to compare two strings.
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<process.h>
void main()
{char str1[10],str2[10];
 int l1=0,l2=0,i,flag=0;
 clrscr( );
 cout<<"\nEnter the first string: ";
 gets(str1);
 cout<<"\nEnter the second string: ";
 gets(str2);
 for(i=0;str1[i]!='\0';i++);
 l1=i;
 for(i=0;str2[i]!='\0';i++);
 l2=i;
 if(l1!=l2)
 {   cout<<"\nBoth strings are not the same...";
     getch( );
     exit(0);
 }
 for(i=0;str1[i]!='\0';i++)
 {   if(str1[i]!=str2[i])
     {  flag=1;
          break; }
 }
 if(flag= =0)
   cout<<"\nEntered strings are identicle...";
 else
   cout<<"\nEntered strings are not identicle...";
 getch( );
}
```

8. Write the output of the following program. 3

```
#include<iostream.h>
int func(int &x,int y=10)
{   if (x%y= = 0)  return ++x;
       Else return y- -; }
 void main( )
{ int p=20,q=23;
  q=func(p,q); cout<<p<<q<<endl;
  p=func(q);cout<<p<<q<<endl;
  q=func(p); cout<<p<<q<<endl; }
```

A. 2023
1023
1111

9.Write a program to print the reverse of a string without using strrev( ) function.

ARRAYS

1.Write a program to print the diagonal (left & right) elements of an N×N matrix. 4

A)
```
//Program to print the left and right diagonal element of an NXN matrix
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
void main( )
{  int A[10][10];
   int i,j,N;
   clrscr( );
   cout<<"\nHow many rows and columns required for matrix: ";
   cin>>N;
   cout<<"\nEnter "<<N*N<<" elements: ";
   for(i=0;i<N;i++)
   {  cout<<"Enter the elements into Row "<<i+1<<": ";
     for(j=0;j<N;j++)
          cin>>A[i][j];
   }
   clrscr( );
   cout<<"\nThe entered elements in the matrix are: \n";
   for(i=0;i<N;i++)
   {  for(j=0;j<N;j++)
          cout<<A[i][j]<<"\t";
     cout<<endl;
   }
   cout<<"\n\n\nThe elements which are belongs to only
 diagonals...\n";
   for(i=0;i<N;i++)
   {  for(j=0;j<N;j++)
          if((i==j)||((i+j)==(N-1)))
             cout<<setw(6)<<A[i][j];
          else
             cout<<"      ";
     cout<<endl;
   }
   getch( );
}
```

2.Write a program to sort an array on N numbers in ascending order. Avoid duplication of elements. 3

A)
```
#include<iostream.h>
#include<conio.h>
void main( )
{      clrscr( );
       int A[20],N,i,j,temp;
       cout<<"\nEnter the number of elements:";
       cin>>N;
       for(i=0;i<N;i++)
            cin>>A[i];
         //Bubble sort technique
         for(i=0;i<N;++i)
              for(j=0;j<(N-1)-i ;j++)
                   if(A[j]>A[j+1])
                    {              Temp=A[j];
                                   A[j]=A[j+1];
                                   A[j+1]=Temp;
                     }
           cout<<"The Elements in the array after sorting…. ";
           for(i=0;i<N;i++)
               cout<<A[i]<<'\t';
}
```

3. Write a program to transpose the matrix of size M*N?3

```
//program to find transpose of M*N Matrix
#include<iostream.h>
#include<conio.h>
void main( )
{ int A[10][10],B[10][10];
  int M,N,i,j;
  cout<<"\nEnter the number of Rows: ";
  cin>>M;
  cout<<"\nEnter the number of Columns: ";
  cin>>N;
  cout<<"\nEnter "<<M*N<<" elements...\n";
  for(i=0;i<M;i++)
    for(j=0;j<N;j++)
    {     cin>>A[i][j];
          B[j][i]=A[i][j];
    }
  cout<<"\nThe entered Array.....\n";
  for(i=0;i<M;i++)
  {  for(j=0;j<N;j++)
          cout<<A[i][j]<<"\t";
    cout<<endl;
  }
  cout<<"\nThe transpose of the matrix.....\n";
  for(i=0;i<N;i++)
  {  for(j=0;j<M;j++)
          cout<<B[i][j]<<"\t";
    cout<<endl;
  }
  getch( );
}
```

4.Write a program to find the maximum number in a given array. 4

For example, if the array elements are 3 5 6 9 12
Max Number = 12

5. What is searching? Explain Sequential Searching with suitable example. 4

6. Write a program in C++ to print sum of diagonals of a 3X3 matrix.

7.Write a program to print the multiplication of two (3 X 3) matrices.

STRUCTURES

1.Rewrite the following program after removing the syntactical error(s) if any. Underline each correction. 2

```
#include<iostream.h>
int main
{ struct movie
{ char movie_name[20];
char movie_type;
int ticket_cost=100;
}movie;
gets(movie_name);
gets(movie_type);
}
```

A)
```
#include<iostream.h>
#include<stdio.h>
void main( )
{    struct movie
     {   char movie_name[20];
       char movie_type;
       int ticket_cost;
    }Movie;
    Movie.ticket_cost=100;
    gets(Movie.movie_name);
    cin>>Movie.movie_type;
    }
```

Very Important Note:
Dear Student, this material is meant only for slow learners to give an idea of questions pattern, collected from 3 old question papers only. Students are advised to read entire syllabus.