LSD DAV PUBLIC SCHOOL, PILKHUWA

D.A.V. COLLEGE MANAGING COMMITTEE

ओ३म्

ESTD. 1886

STUDY MATERIAL

CLASS-XII

COMPUTER SCIENCE (083)

SESSION 2017-18

Rajeshwar Tomar : MCA, BCA

PGT computer science

PREFACE

It gives me immense pleasure to present the Study Material of Class XII Computer Science (083) for session 2017-18 by LSD DAV PUBLIC SCHOOL, PILKHUWA.

This study material is written according to CBSE Syllabus of Computer Science (083) for Class XII.

I am confident that the Study Material for Class XII Computer Science will help the students immensely to understand the concepts and will improve the quality performance of the students.

Wish you all the best.

(RAJESHWAR TOMAR)

PGT COMPUTER SCIENCE

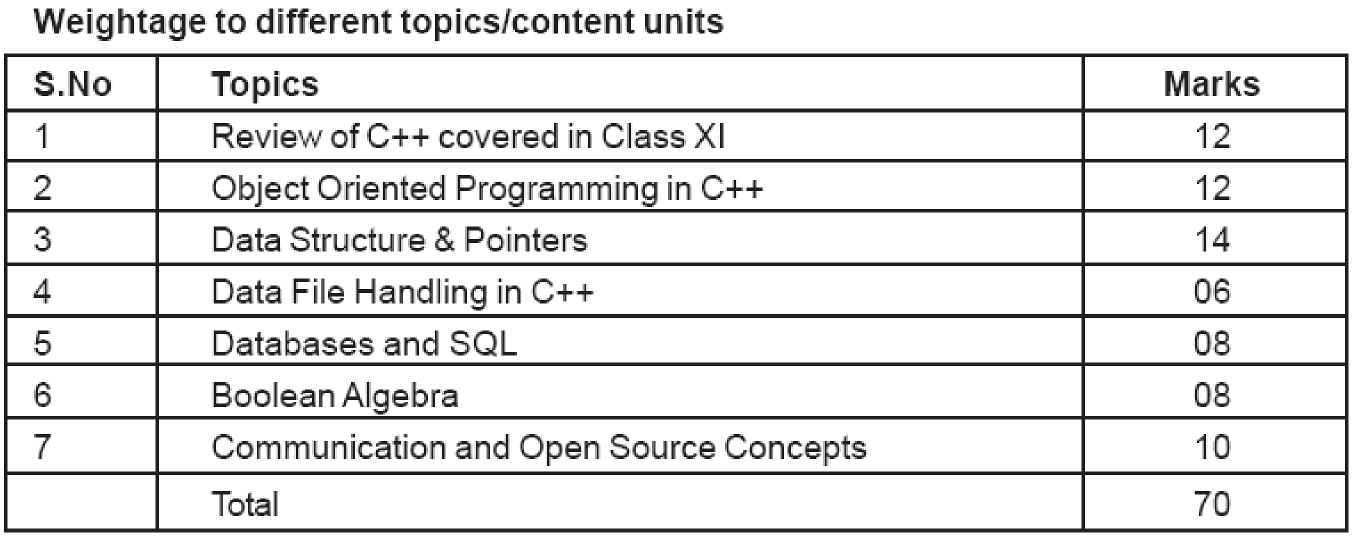
LSD DAV PUBLIC SCHOOL

TIPS FOR STUDENTS

1. Prepare those questions first, which appear easy to you.
2. Memorizing Important terms of a topic will help immensely.
3. Practice each answer/solution in writing apart from reading.
4. Practice all similar type of questions at a time.
5. Read the questions carefully, before answering.
6. To leave good impressions on Examiner attempt those questions first in which you are very much confident.
7. Don't stretch the answer unnecessarily.
8. Try to write answer in points.
9. Important point should be underlined but be careful, don't waste your time.
10. Try to illustrate your answer graphically, if possible.
11. Don't leave any question unanswered.
12. Solve at-least 05 previous years question papers.
13. Make precise and concise notes, point wise for exam time preparation/quick revision.
14. Plan your study judiciously.
15. A proper timetable for study should be followed strictly.
16. Take healthy and timely diet during examinations. Also take sound sleep every day.
17. Take regular breaks in each study duration.
18. Do not forget to revise all the topics one day prior, to the day of examination.
19. Take good care of your health.
20. Check CBSE Academic website (http://www.cbseacademic.in/) regularly for updates and for support materials like e-books, e-contents etc. visit http://kvspgtcs.org/

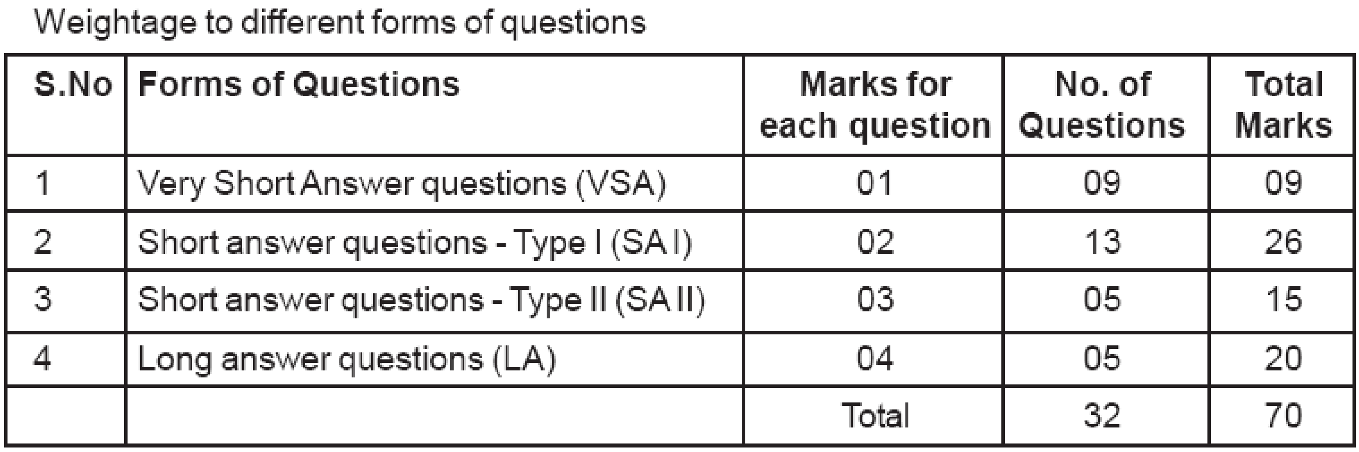
Unit Wise Marks Distribution in QP for last six years

S. No	Unit Name	Marks
1	UNIT 1 Programming in C++	30
2	UNIT 2 Data structures	14
3	UNIT 3 Database and SQL	08
4	UNIT 4 Boolean Logic	08
5	UNIT 5 Communication and open source concept	10
	Total Marks	70



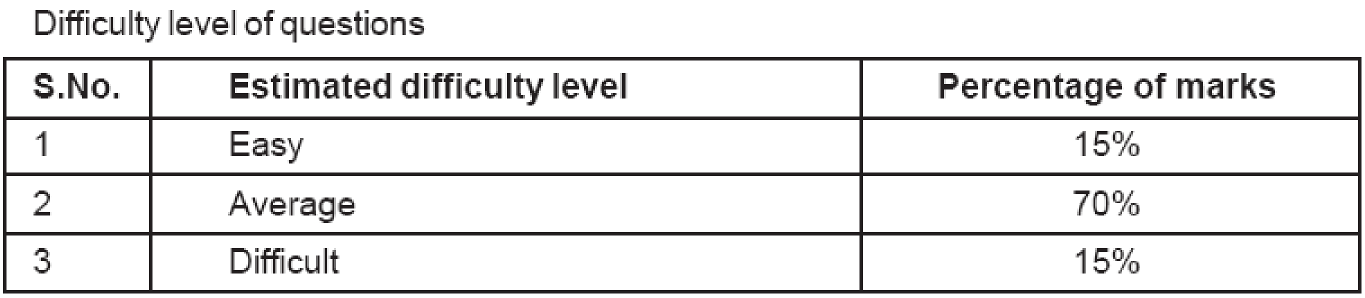
Weightage to different topics/content units

S.No	Topics	Marks
1	Review of C++ covered in Class XI	12
2	Object Oriented Programming in C++	12
3	Data Structure & Pointers	14
4	Data File Handling in C++	06
5	Databases and SQL	08
6	Boolean Algebra	08
7	Communication and Open Source Concepts	10
	Total	70



Weightage to different forms of questions

S.No	Forms of Questions	Marks for each question	No. of Questions	Total Marks
1	Very Short Answer questions (VSA)	01	09	09
2	Short answer questions - Type I (SA I)	02	13	26
3	Short answer questions - Type II (SA II)	03	05	15
4	Long answer questions (LA)	04	05	20
		Total	32	70



Difficulty level of questions

S.No.	Estimated difficulty level	Percentage of marks
1	Easy	15%
2	Average	70%
3	Difficult	15%

UNIT 1 : PROGRAMMING IN C++

Introduction to C++

- C++ programming language developed by AT&T Bell Laboratories in 1979 by Bjarne Stroustrup. C++ is fully based on Object Oriented Technology i.e. C++ is ultimate paradigm for the modeling of information.
- C++ is the successor of C language.
- It is a case sensitive language.

Character Set- Set of valid characters which are recognized by c++compiler i.e Digits (0-9), Alphabets (A-Z & a-z) and special characters + - * , . " ' < > = { ( ]) space etc i.e 256 ASCII characters.

Tokens - Smallest individual unit. Following are the tokens

Keyword-Reserve word having special meaning the language and can't be used as identifier.

- **Identifiers-** Names given to any variable, function, class, union etc. Naming convention (rule) for writing identifier is as under:
  (i) It can contain Alphabets, digits or underscore
  (ii) It must be started with alphabet or underscore (_).
  (iii) It must not contain special characters.
  (iv) It cannot be a Reserve word.

- **Literals-**Value of specific data type assign to a variable or constant. Four type of Literals:
  i. Integer Literal i.e int x =10
  ii. Floating point Literal i.e float x=123.45
  iii. Character Literal i.e char x= 'a', single character enclosed in single quotes.
  iv. String Literal i.e cout<< "Welcome" , anything enclosed in double quotes

- Operator – performs some action on data
- Arithmetic(+,-,*,/,%)
- Assignment operator (=)
- Increment / Decrement (++, --)
- Relational/comparison (<,>,<=,>=,==,!=).
- Logical(AND(&&),OR(||),NOT(!).
- Conditional (? :)

Precedence of operators:

++(post increment),--(post decrement)	High
++(pre increment),--(pre decrement),sizeof !(not),-(unary),+unary plus)	
*(multiply), / (divide), %(modulus)	
+(add),-(subtract)	
<(less than), <=(less than or equal), >(greater than), >=(greater than or equal to) ==(equal), !=(not equal)	
&& (logical AND) || (logical OR)	
?:(conditional expression)	
=(simple assignment) and other assignment operators(arithmetic assignment operator)	
, Comma operator	Low

- Punctuation – used as separators in c++ e.g. [{ ( ) }] , ; # = : etc

Data type- A specifier to create memory block of some specific size and type. C++offers two types of data types:

1. **Fundamental type :** Which are not composed any other data type i.e. int, char, float and void
2. **Derived data type:** Which are made up of fundamental data type i.e array, function, class, union etc

Data type conversion- Conversion of one data type into another data type. Two type of conversion

i. **Implicit Conversion –** It is automatically taken care by complier in the case of lower range to higher range e.g. int x, char c='A' then x=c is valid i.e character value in c is automatically converted to integer.

ii. **Explicit Conversion-** It is user-defined that forces an expression to be of specific type. e.g. double x1,x2 and int res then res=int(x1+x2)

Variable- Named storage location where value can be stored and changed during program execution. e.g. **int x, float y, float amount, char c;**

Constant- Memory block where value can be stored once but can't be changed later on during program execution. e.g. **const int pi =3.14;**

cout – It is an object of **ostream class** defined in **iostream.h** header file and used to display value on monitor.

cin – It is an object of **istream** class defined in **iostream.h** header file and used to read value from keyboard for specific variable.

comment- Used for better understanding of program statements and escaped by the compiler to compile . e.g. – single line (//) and multi- line(/*….*/)

Cascading – Repeatedly use of input or output operators(">>" or "<<") in one statement with cin or cout.

Control structure:

Sequence control statement(if)	conditional statement (if else)	Multiple Choice Statement If –else-if	Switch Statement (Alternate for ifelse-if) works for only exact match	loop control statement (while ,do… while, for)
Syntax	Syntax	Syntax	Syntax	Syntax

if(expression)
{
statements;
}
	If(expression)
	{
	statements;
	}
	else
	{
	statements;
	}
		If (expression)
		{
		statements
		}
		else if(expression)
		{
		statement
		}
		else
		{
		statement
		}
			switch(int / char variable)
			{ case literal1:
			[statements
			break;]
			case literal2:
			[statements,
			break;]
			default:statements;
			}
			Break is compulsory statement with every case because if it is not included then the controls
				while(expression)
				{
				statements;
				}
				Entry control loop works for true condition.
				
				do
				{
				statements;
				} while(expression);
				Exit Control Loop
				
				execute at least once if the condition is false at beginning.

			executes next case statement until next break encountered or end of swtich reached.
			Default is optional, it gets executed when no match is found
				for loop
				for(expression1;expression2;expression3)
				{
				statement;}
				Entry control loop works for true condition and preferred for fixed no.of times.

Note: any non-zero value of an expression is treated as true and exactly 0 (i.e. all bits contain 0) is treated as false.

Nested loop -loop within loop.
exit()- Defined in process.h and used to terminate the program depending upon certain condition.
break- Exit from the current loop depending upon certain condition.
continue- to skip the remaining statements of the current loop and passes control to the next loop control statement.
goto- control is unconditionally transferred to the location of local label specified by <identifier>.

```
For example
A1:
cout<<”test”;

goto A1;
```

Some Standard C++ libraries

Header File	Purpose
iostream.h	Defines stream classes for input/output streams
stdio.h	Standard input and output
ctype.h	Character tests
string.h	String operations
math.h	Mathematical functions such as sin() and cos()
stdlib.h	Utility functions such as malloc() and rand()

Some functions

- isalpha(c)- check whether the argument is alphabetic or not.
- islower(c)- check whether the argument is lowecase or not.
- isupper(c) - check whether the argument is upercase or not.
- isdigit(c)- check whether the argument is digit or not.
- isalnum(c)- check whether the argument is alphanumeric or not.
- tolower()- converts argument in lowercase if its argument is a letter.
- toupper(c)- converts argument in uppercase if its argument is a letter.

CYTYPE. H

- strcat()- concatenates two string.
- strcmp- compare two string.

STRING.H

- pow(x,y)- return x raised to power y.
- sqrt(x)- return square root of x.

MATHS.H

- random(num)-return a random number between 0 and (num-1)
- randomize- initializes the random number generator with a random value.

STDLIB.H

iomanip (iomanip.h)

The manipulation in this library affect the format of steam operations. Note that iostream contains additional manipulators.

setbase(b)	Setts number base to b = 8, 10, or 16
setfill(f)	Sets fill character to f
setprecision(n)	Sets floating-point precision to integer n
setw(n)	Sets field width to integer n

Array- Collection of element of same type that are referred by a common name.

One Dimensional array

- An array is a continuous memory location holding similar type of data in single row or single column. Declaration in c++ is as under:
  const int size =10;
  int a[size] or int a[20]. The elements of array accessed with the help of an index.
  For example : for(i=0;i<20;i++) cout<<a[i];

A	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]

- **String (Array of characters)** –Defined in c++ as one dimensional array of characters as char s[80]= “KV Kumbhirgram”;

S	K	V		K	u	m	b	h	i	r	g	r	a	m	\0					

Two dimensional array

A two dimensional array is a continuous memory location holding similar type of data arranged in row and column format (like a matrix structure).

Declaration – int a[3][4], means ‘a’ is an array of integers are arranged in 3 rows & 4columns.

	0	1	2	3
0	A[0][0]	A[0][1]	A[0][2]	A[0][3]
1	A[1][0]	A[1][1]	A[1][2]	A[1][3]
2	A[2][0]	A[2][1]	A[2][2]	A[2][3]

Function -Name given to group of statements that does some specific task and may return a value. Function can be invoked (called) any no. of time and anywhere in the program.

Function prototypes-Function declaration that specifies the function name, return type and parameter list of the function.

syntax: **return_type** function_name (**type var1, type var2,…., type varn**) **;**

Actual Parameters

Variables associated with function name during function call statement.

Formal Parameters

Variables which contains copy of actual parameters inside the function definition.

Local variables

Declared inside the function only and its scope and lifetime is function only and hence accessible only inside function.

Global variables

Declared outside the function and its scope and lifetime is whole program and hence accessible to all function in the program from point declaration.

Example :

```
#include <iostream.h>
int a=20; // global
void main()
{
        int b=10; // local
        cout<<a<<b;
}
```

Passing value to function-

- **Passing by value-** In this method separate memory created for formal arguments and if any changes done on formal variables, it will not affect the actual variables. So actual variables are preserved in this case
- **Passing by address/reference-** In this method no separate memory created for formal variables i.e formal variables share the same location of actual variables and hence any change on formal variables automatically reflected back to actual variables.

Example :

```
void sample( int a, int &b)
{      a=a+100;
       b=b+200;
       cout<<a<<b;
}
void main()
{           int a=50, b=40;
            cout<<a<<b; // output 50 40
            sample(a,b) // output 150 240
            cout<<a<<b; // output 50 240
}
```

Function overloading

- Processing of two or more functions having same name but different list of parameters

Function recursion

- Function that call itself either directly or indirectly.

Structure-Collection of logically related different data types (Primitive and Derived) referenced under one name.

```
e.g. struct employee
        {
                int empno;
                char name[30];
                char design[20];
                char department[20];
        }
```

Declaration: employee e;

```
Input /Output : cin>>e.empno; // members are accessed using dot(.) operator.
                 cout<<e.empno;
```

Nested structure

- A Structure definition within another structure.
- A structure containing object of another structure.

```
e.g. struct address
        {
                int houseno;
                char city[20];
                char area[20];
                long int pincode;
        }
        struct employee
        {
                int empno;
                char name[30];
                char design[20];
                char department[20];
                address add; // nested structure
        }
```

Declaration: employee e;

Input /Output : **cin>>e.add.houseno;** // members are accessed using dot(.) operator.
cout<<e.ad.houseno;

typedef-Used to define new data type name.

e.g. typedef char Str80[80]; Str80 str;

#define Directives

- Use to define a constant number or macro or to replace an instruction.

1 Marks questions

(1) **Which C++ header file(s) will be essentially required to be included to run /execute the following C++ code:**

```
void main()
{
        char Msg[ ]="Sunset Gardens";
        for (int I=5;I<strlen(Msg);I++)       //String.h
        puts(Msg);                            // stdio.h
}
```

Ans : stdio.h, string.h

(2) **Name the header files that shall be need for the following code: (CBSE 2012)**

```
void main()
{
        char text[] =”Something”
        cout<<”Remaining SMS chars: ”<<160-strlen(text)<<endl; //string.h
}
```

Ans: iostream.h, string.h

2 Marks questions:

1) **Rewrite the following program after removing the syntactical error(s) if any.Underline each correction**. **CBSE 2012**

```
#include<iostream.h>
Class Item
{
     long IId, Qty;
public:
     void Purchase { cin>>IId>>Qty;}
     void Sale()
     {
             cout<<setw(5)<<IId<<”Old:”<< Qty<<endl;
             cout<< “New :”<<Qty<<endl;
     }   };

void main()
{
     Item I;
     Purchase();
     I.Sale()
}
```

Ans : #include<iostream.h>

```
class Item // C capital
{
long IId, Qty;
public:
void Purchase ( ) { cin>>IId>>Qty;} // ( ) after function name
void Sale( )
{
cout<<setw(5)<<IId<<"Old:"<< Qty<<endl;
cout<< "New :"<<Qty<<endl;
}};
void main()
{
Item I;
I. Purchase( ); // object missing
I.Sale( ) ; // ; is missing
}
```

2) Find the output of the following program: CBSE 2012

```
#include<iostream.h>
#include<ctype.h>
typedef char Str80[80];
void main()
{
        char *Notes;
        Str80 str= " vR2GooD";
        int L=6;
        Notes =Str;
        while(L>=3)
        {
              Str[L]=(isupper(Str[L])? tolower(Str[L]) : toupper(Str[L]));
              cout<<Notes<<endl;
              L--;
              Notes++;
        }
  }
```

*** consider all required header file are include**

Note (index)	**L**	**str**	**Output**
		vR2GooD	
0	**6**	vR2Goo**d**	vR2Goo**d**
1	**5**	vR2Go**O**d	R2Go**O**d
2	**4**	vR2G**O**Od	2G**O**Od
3	**3**	vR2**g**OOd	**g**OOd

Ans : vR2Good

```
R2GoOd
2GOOd
gOOd
```

3) Observe the following program and find out, which output(s) out id (i) to (iv) will not be expected from program? What will be the minimum and maximum value assigned to the variables Chance?

```
#include<iostream.h> CBSE 2012
#include<stdlib.h>
```

```
void main()
{
    randomize();
    int Arr[] = {9,6};, N;
    int Chance = random(2)+10;
    for(int c=0;c<2;c++)
    {
            N= random(2);
            cout<<Arr[N];
    }
}
```

i) 9#6#
ii) 19#17#
iii) 19#16#
iv) 20#16#

Ans: The output not expected from program are (i),(ii) and (iv)
Minimum value of Chance =10
Maximum value of Chance = 11

3 Marks questions:

4) Find the output of the following program: CBSE 2012

```
#include<iostream.h>
class METRO
{
    int Mno, TripNo, PassengerCount;
public:
    METRO ( int Tmno=1)
    { Mno =Tmno; PassengerCount=0;}
    void Trip(int PC=20)
    { TripNo++, PassengerCount+=PC};
    void StatusShow()
    { cout<<Mno<< ":"<<TripNo<< " :"<<PassengerCount<<endl;}
};
void main()
{
```

M

5	0	0
5	**1**	**20**

```
    METRO M(5),T;
    M.Trip();
```

T

1	0	0
1	**1**	**0**

```
     M.StatusShow();
    T.StatusShow();
    M.StatusShow();
}
```

Ans : 5: 1: 20
1: 1: 0
5: 1: 20

2& 3 marks practice questions:

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

```
#include<iostream.h>
void main( )
{ F = 10, S = 20;
test(F;S);
test(S);
}
void test(int x, int y = 20)
{ x=x+y;
count<<x>>y;
}
```

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

```
#include “iostream.h”
Class MEMBER
{ int Mno;
float Fees;
PUBLIC:
void Register ( ) {cin>>Mno>>Fees;}
void Display( ) {cout<<Mno<<" : "<<Fees<<endl;}
};
void main()
{ MEMBER delete;
Register();
delete.Display();
}
```

7) Find the output for the following program:

```
#include<iostream.h>
#include<ctype.h>
void Encript ( char T[ ])
{ for( int i=0 ; T[i] != ‘ \0’ ; i += 2)
if( T[i] = = ‘A’ || T[i] = = ‘E’ )
T[i] = ‘#’ ;
else if (islower (T[i] ))
T[i] = toupper(T[i]);
else
T[i] = ‘@’;}
```

```
void main()
{ char text [ ] = "SaVE EArTh in 2012";
encrypt(text);
cout<<text<<endl;
}
```

8) Find the output of the following program:

```
#include<iostream.h>
void main( )
{ int U=10,V=20;
for(int I=1;I<=2;I++)
{ cout<<"[1]"<<U++<<"&"<<V 5 <<endl;
cout<<"[2]"<<++V<<"&"<<U + 2 <<endl; } }
```

9) **Rewrite the following C++ program after removing the syntax error(s) if any. Underline each correction. [CBSE 2010]**

```
include<iostream.h>
class FLIGHT
{ Long FlightCode;
Char Description[25];
public
void addInfo()
{ cin>>FlightCode; gets(Description);}
void showInfo()
{ cout<<FlightCode<<":"<<Description<<endl;}
};
void main( )
{ FLIGHT F;           addInfo.F();    showInfo.F;    }
```

10) In the following program, find the correct possible output(s)from the options:

```
#include<stdlib.h>
#include<iostream.h>
void main( )
{ randomize( );
char City[ ][10]={"DEL", "CHN", "KOL", "BOM", "BNG"};
int Fly;
for(int I=0; I<3;I++)
{Fly=random(2) + 1;
cout<<City[Fly]<< ":";
}}
```

Outputs:

(i) DEL : CHN : KOL: (ii) CHN: KOL : CHN:

(iii) KOL : BOM : BNG: (iv) KOL : CHN : KOL:

11) In the following C++ program what is the expected value of Myscore from options (i) to (iv) given below. Justify your answer.

```
#include<stdlib.h>
#include<iostream.h>
void main( )
{ randomize( );
int Score[ ] = {25,20,34,56,72,63},Myscore;
cout<<Myscore<<endl;
}
```

i) 25 (ii) 34 (iii) 20 (iv) Garbage Value.

Function overloading in C++

- A function name having several definitions that are differentiable by the number or types of their arguments is known as **function overloading**.

Example : A same function **print()** is being used to print different data types:

```
#include <iostream.h>
class printData
{
public:
void print(int i) {
    cout << "Printing int: " << i << endl;
}
void print(double f) {
    cout << "Printing float: " << f << endl;
}
void print(char* c) {
    cout << "Printing character: " << c << endl;
}
};

int main(void)
{
    printData pd;   // Call print to print integer
    pd.print(5);      // Call print to print float
    pd.print(500.263);//// Call print to print character
    pd.print("Hello C++"); return 0;         }
```

When the above code is compiled and executed, it produces following result:

```
Printing int: 5
Printing float: 500.263
Printing character: Hello C++
```

OBJECT ORIENTED PROGRAMMING CONCEPTS

Object Oriented Programming follows bottom up approach in program design and emphasizes on safety and security of data..

FEATURES OF OBJECT ORIENTED PROGRAMMING:

Inheritance:

- Inheritance is the process of forming a new class from an existing class or base class. The base class is also known as parent class or super class.
- Derived class is also known as a child class or sub class. Inheritance helps in reusability of code , thus reducing the overall size of the program

Data Abstraction:

- It refers to the act of representing essential features without including the background details .Example : For driving , only accelerator, clutch and brake controls need to be learnt rather than working of engine and other details.

Data Encapsulation:

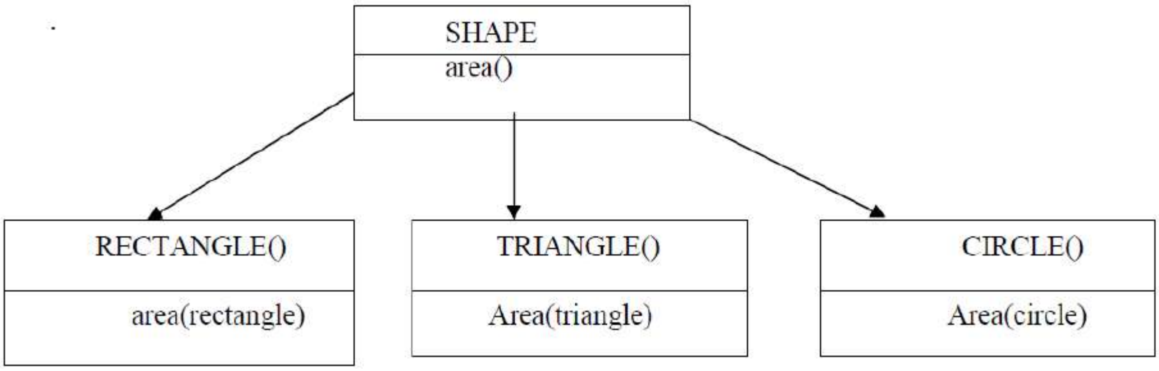
- It means wrapping up data and associated functions into one single unit called class..
- A class groups its members into three sections: public, private and protected, where private and protected members remain hidden from outside world and thereby helps in implementing data hiding.

Modularity :

- The act of partitioning a complex program into simpler fragments called modules iscalled as modularity.
- It reduces the complexity to some degree and
- It creates a number of well-defined boundaries within the program .

Polymorphism:

- **Poly** means many and **morphs** mean form, so polymorphism means one name multiple forms.
- It is the ability for a message or data to be processed in more than one form.
- C++ implements Polymorhism through Function Overloading , Operator overloading and Virtual functions .



Objects and Classes:

The major components of Object Oriented Programming are. **Classes & Objects**

A **Class i**s a group of similar objects. **Objects** share two characteristics *state* and *behavior (data members and member functions)*

Classes in Programming :

- **It is a collection of variables, often of different types and its associated functions.**
- **Class just binds data and its associated functions under one unit there by enforcing**

encapsulation.

- Classes define types of data structures and the functions that operate on those data structures.
- A class defines a blueprint for a data type.

Declaration/Definition :

A class definition starts with the keyword **class** followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations.

```
class class_name
{
access_specifier_1:
        member1;
access_specifier_2:
        member2;
...
} object_names;
```

[Note: the default access specifier is private.

Example :

```
class Box
{
          int a;
public:
         double length;          // Length of a box
         double breadth;        // Breadth of a box
         double height;         // Height of a box
} container;                    // container is an object of class Box
```

Access specifiers in Classes:

Access specifiers are used to identify access rights for the data and member functions of the class.

There are three main types of access specifiers in C++ programming language:

- private
- public
- protected

Member-Access Control

Type of Access	**Meaning**
Private	Class members declared as **private** can be used only by member functions and friends (classes or functions) of the class.
Protected	Class members declared as **protected** can be used by member functions and friends (classes or functions) of the class. Additionally, they can be used by classes derived from the class.
Public	Class members declared as **public** can be used by any function.

- **Importance of Access Specifiers**

Access control helps prevent you from using objects in ways they were not intended to be used. Thus it helps in implementing data hiding and data abstraction.

OBJECTS in C++:

Objects represent instances of a class. Objects are basic run time entities in an object oriented system.

Creating object / defining the object of a class:

The general syntax of defining the object of a class is:-

Class_name object_name;

In C++, a class variable is known as an object. The declaration of an object is similar to that of a variable of any data type. The members of a class are accessed or referenced using object of a class.

```
    Box Box1; // Declare Box1 of type Box
    Box Box2; // Declare Box2 of type Box
```

Both of the objects Box1 and Box2 will have their own copy of data members.

Accessing / calling members of a class. ***All member of a class are private by default***.

- Private member can be accessed only by the function of the class itself.
- Public member of a class can be accessed through any object of the class. They are accessed or called using object of that class with the help of dot operator (.).

The general syntax for accessing data member of a class is:-

Object_name.Data_member = value;

The general syntax for accessing member function of a class is:-

Object_name. Function_name (actual arguments);

Class methods definitions (Defining the member functions)

Member functions can be defined in two places:-

- ### Outside the class definition

The member functions of a class can be defined outside the class definitions. It is only declared inside the class but defined outside the class. The general form of member function definition outside the class definition is:

```
Return_type Class_name:: function_name (argument list)
{
    Function body
}
```

Where symbol :: is a scope resolution operator.

```
class sum
{
    int A, B, Total;
public:
    void getdata ();
    void display ();
};
void sum:: getdata () // Function definition outside class definition Use of :: operator
{
    cout<<" \ n enter the value of Aand B";
    cin>>A>>B;
}
void sum:: display () // Function definition outside class definition Use of :: operator
{
    Total =A+B;
    cout<<"\n the sum of A and B="<<Total;
}
```

- **Inside the class definition**

The member function of a class can be declared and defined inside the class definition.

```
class sum
{
        int A, B, Total;
public:
        void getdata ()
        {
                cout< "\n enter the value of A and B";
                cin>>A>>B;
        }
        void display ()
        {
                total = A+B;
                cout<<"\n the sum of A and B="<<total;
        }
};
```

Differences between struct and classes in C++

Structure	Class
• **Defined using struct keyword** • **Members are public by default** • **It cannot be inherited**	• **Defined using class keyword** • **Members are private by default** • **It can be inherited**

INLINE FUNCTIONS

- **Inline functions definition starts with keyword inline**
- **The compiler replaces the function call statement with the function code itself(expansion) and then compiles the entire code.**
- **They run little faster than normal functions as function calling overheads are saved.**
- **A function can be declared inline by placing the keyword inline before it.**

Example

```
inline void Square (int a)
{
```

In place of function call , function body is substituted because Square () is inline function

```
        cout<<a*a;
```

```
}
void main()
{.
        Square(4); { cout <<4*4;}               // { cout <<4*4;}
        Square(8) ; { cout <<8*8; }             // { cout <<8*8; }
}
```

Pass Object As An Argument

/*C++ PROGRAM TO PASS OBJECT AS AN ARGUMEMT. The program Adds the two heights given in feet and inches. */

```
#include< iostream.h>
#include< conio.h>
class height
{
        int feet,inches;
public:
        void getht(int f,int i)
        {
                feet=f;
                inches=i;
        }
        void putheight()
        {
                cout< < "\nHeight is:"< < feet< < "feet\t"< < inches< < "inches"< < endl;
        }
        void sum(height a,height b)
        {
                height n;
                n.feet = a.feet + b.feet;
                n.inches = a.inches + b.inches;
                if(n.inches ==12)
                {
                        n.feet++;
                        n.inches = n.inches -12;
                }
                cout< < endl< < "Height is "< < n.feet< < " feet and "< < n.inches< < endl;
        }
};
void main()
{
        height h,d,a;
        clrscr();
        h.getht(6,5);
        a.getht(2,7);
        h.putheight();
        a.putheight();
        d.sum(h,a);
        getch();
}
```

```
/**********OUTPUT***********
Height is 6feet 5inches
Height is 2feet 7inches
Height is 9 feet and 0
```

4 Marks Solved Problems:

Q 1**) Define a class TAXPAYER in C++ with following description:**

Private members :

- Name of type string
- PanNo of type string
- Taxabincm (Taxable income) of type float
- TotTax of type double
- A function CompTax( ) to calculate tax according to the following slab:

Taxable Income	**Tax%**
Up to 160000	0
>160000 and <=300000	5
>300000 and <=500000	10
>500000	15

Public members :

- A parameterized constructor to initialize all the members
- A function INTAX( ) to enter data for the tax payer and call function CompTax( ) to assign TotTax.
- A function OUTAX( ) to allow user to view the content of all the data members.

Ans.

```
class TAXPAYER
{
        char Name[30],PanNo[30];
        float Taxabincm;
        double TotTax;
        void CompTax()
        {
                if(Taxabincm >500000)
                        TotTax= Taxabincm*0.15;
                else if(Taxabincm>300000)
                        TotTax= Taxabincm*0.1;
                else if(Taxabincm>160000)
                        TotTax= Taxabincm*0.05;
                else
                        TotTax=0.0;
        }
public:
        TAXPAYER(char nm[], char pan[], float tax, double tax) //parameterized constructor
        {
                strcpy(Name,nm);
                strcpy(PanNo,pan);
                Taxabincm=tax;
                TotTax=ttax;
        }
        void INTAX()
        {
                gets(Name);
                cin>>PanNo>>Taxabincm;
                CompTax();
        }
        void OUTAX()
        { cout<<Name<<'\n'<<PanNo<<'\n'<<Taxabincm<<'\n'<<TotTax<<endl; }
};
```

Q 2 : **Define a class HOTEL in C++ with the following description:**

Private Members

- Rno //Data Member to store Room No
- Name //Data Member to store customer Name
- Tariff //Data Member to store per day charge
- NOD //Data Member to store Number of days
- CALC //A function to calculate and return amount as NOD*Tariff and if the value of NOD*Tariff is more than 10000 then as 1.05*NOD*Tariff

Public Members:

- Checkin( ) //A function to enter the content RNo,Name, Tariff and NOD
- Checkout() //A function to display Rno, Name, Tariff, NOD and Amount (Amount to be displayed by calling function CALC( )

Solution :

```
#include<iostream.h>
class HOTEL
{
        unsigned int Rno;
        char Name[25];
        unsigned int Tariff;
        unsigned int NOD;
        int CALC( )
        {
                 int x;
                x=NOD*Tariff;
                if( x>10000)
                        return(1.05*NOD*Tariff);
                else
                        return(NOD*Tariff);
        }
public:
        void Checkin()
        {
                cin>>Rno>>Name>>Tariff>>NOD;
        }
        void Checkout()
        {
                cout<<Rno<<Name<<Tariff<<NOD<<CALC();
        }
};
```

Q 3 Define a class Applicant in C++ with following description:

Private Members

- **A data member ANo (Admission Number) of type long**
- **A data member Name of type string**
- **A data member Agg(Aggregate Marks) of type float**
- **A data member Grade of type char**
- **A member function GradeMe( ) to find the Grade as per the Aggregate Marks obtained by a student. Equivalent Aggregate marks range and the respective Grades are shown as follows**

Aggregate Marks	Grade
> = 80	A
Less than 80 and > = 65	B
Less than 65 and > = 50	C
Less than 50	D

Public Members

- **A function Enter( ) to allow user to enter values for ANo, Name, Agg & call function GradeMe( ) to find the Grade**
- **A function Result ( ) to allow user to view the content of all the data members.**

Ans:

```
class Applicant
{
        long ANo;
        char Name[25];
        float Agg;
        char Grade;
        void GradeMe( )
        {
                 if (Agg > = 80)
                        Grade = ‘A’;
                else if (Agg >= 65 && Agg < 80 )
                        Grade = ‘B’;
                else if (Agg >= 50 && Agg < 65 )
                        Grade = ‘C;
                else
                        Grade = ‘D’;
        }
public:
        void Enter ( )
        {
                cout <<”\n Enter Admission No. “; cin>>ANo;
                cout <<”\n Enter Name of the Applicant “; cin.getline(Name,25);
                cout <<”\n Enter Aggregate Marks obtained by the Candidate :“; cin>>Agg;
                GradeMe( );
        }
        void Result( )
        {
                cout <<”\n Admission No. “<<ANo;
                cout <<”\n Name of the Applicant “;<<Name;
                cout<<”\n Aggregate Marks obtained by the Candidate. “ << Agg;
                cout<<\n Grade Obtained is “ << Grade ;
        }
};
```

Q 4 Define a class ITEM in C++ with following description:
Private members:

- **Icode of type integer (Item Code)**
- **Item of type string (Item Name)**
- **Price of type Float (Price of each item)**
- **Qty of type integer (Quantity in stock)**
- **Discount of type float (Discount percentage on the item)**
- **A find function finddisc( ) to calculate discount as per the following rule:**

If Qty <=50	**discount is 0%**
If 50 < Qty <=100	**discount is 5%**
If Qty>100	**discount is 10%**

Public members :

- **A function Buy( ) to allow user to enter values for Icode, Item,Price, Qty and call function Finddisc ( ) to calculate the discount.**
- **A function showall ( ) to allow user to view the content of all the data members.**

Ans :

```
class ITEM
{       int Icode,Qty;
        char item[20];
        float price,discount;
        void finddisc();
public:
        void buy();
        void showall();
};
void stock::finddisc( )
{       if (qty<=50)
                Discount=0;
        else if (qty> 50 && qty <=100)
                Discount=0.05*price;
        else if (qty>100)
                Discount=0.10*price;
}
void stock::buy( )
{       cout<<"Item Code :";cin>>Icode;
        cout<<"Name :";gets(Item);
        cout<<"Price :";cin>>Price;
        cout<<"Quantity :";cin>>Qty;
        finddisc();
}
void TEST::DISPTEST()
{       cout<<"Item Code :";cout<<Icode;
        cout<<"Name :";cout<<Item;
        cout<<"Price :";cout<<Price;
        cout<<"Quantity :";cout<<Qty;
        cout<<"Discount :";cout<<discount;
}
```

4 marks Practice Problems :

Q 1 Define a class **employee** with the following specifications : **4**

Private members of class employee

- empno integer
- ename 20 characters
- basic, hra, da float
- netpay float
- calculate() A function to calculate basic + hra + da with float return type

Public member function of class employee

- havedata() function to accept values for empno, sname, basic, hra, da and invoke calculate() to calculate netpay.
- dispdata() function to display all the data members on the screen.

Q2 Define a class **Student** with the following specifications : **4**

Private members :

roll_no	integer
name	20 characters
class	8 characters
marks[5]	integer
percentage	float

- Calculate() a function that calculates overall percentage of marks and return the percentage of marks.

public members :

- Readmarks() a function that reads marks and invoke the Calculate function.
- Displaymarks() a function that prints the marks.

Q3 : Define a class **repor**t with the following specification : **4**

Private members :

adno	4 digit admission number
name	20 characters
marks	an array of 5 floating point values
average	average marks obtained
getavg()	to compute the average obtained in five subjects

Public members :

- readinfo() function to accept values for adno, name, marks, and invoke the function getavg().
- displayinfo() function to display all data members on the screen you should give function definitions.

Q4 Declare a class **myfolder** with the following specification : **4**

Private members of the class

- Filenames – an array of strings of size[10][25](to represent all the names of files inside myfolder)

- Availspace – long (to represent total number of bytes available in myfolder)
- Usedspace – long (to represent total number of bytes used in myfolder)

public members of the class

- Newfileentry() – A function to accept values of Filenames, Availspace and Usedspace fromuser
- Retavailspace() – A Fucntion that returns the value of total Kilobytes available (1 Kilobytes = 1024 bytes)
- Showfiles() – a function that displays the names of all the files in myfolder

2 Marks Practice Problems

1. What is relation between class and object?
2. What are inline functions? Give example
3. Difference between private & public access specifiers.
4. How class implements data-hiding & encapsulation?
5. What is the difference between structure and a class?
6. How is inline function different from a normal function?

CONSTRUCTORS AND DESTRUCTORS

CONSTRUCTORS :

A member function with the same as its class is called Constructor and it is used to initialize the object of that class with a legal initial value.

Example :

```
class Student
{
	int rollno;
	float marks;
public:
	student( )			//Constructor
	{
		rollno = 0 ;
		marks = 0.0 ;
	}
//other public members
};
```

TYPES OF CONSRUCTORS:

1. Default Constructor:

A constructor that accepts no parameter is called the Default Constructor. If you don't declare a constructor or a destructor, the compiler makes one for you. The default constructor and destructor take no arguments and do nothing.

2. Parameterized Constructors:

A constructor that accepts parameters for its invocation is known as parameterized Constructors , also called as Regular Constructors.

DESTRUCTORS:

A destructor is also a member function whose name is the same as the class name but is preceded by tilde(“~”).It is automatically by the compiler when an object is destroyed. Destructors are usually used to deallocate memory and do other cleanup for a class object and its class members when the object is destroyed.

A destructor is called for a class object when that object passes out of scope or is explicitly deleted.

Example :

```
class TEST
{
	int Regno,Max,Min,Score;
Public:
	TEST( ) // Default Constructor
	{			}
	TEST (int Pregno,int Pscore)		// Parameterized Constructor
	{
		Regno = Pregno ;Max=100;Max=100;Min=40;Score=Pscore;
	}
	~ TEST ( )			// Destructor
	{ Cout<<”TEST Over”<<endl;}
};
```

The following points apply to constructors and destructors:

- Constructors and destructors do not have return type, not even void nor can they return values.
- References and pointers cannot be used on constructors and destructors because their addresses cannot be taken.
- Constructors cannot be declared with the keyword virtual.
- Constructors and destructors cannot be declared static, const, or volatile.
- Unions cannot contain class objects that have constructors or destructors.
- The compiler automatically calls constructors when defining class objects and calls destructors when class objects go out of scope.
- Derived classes do not inherit constructors or destructors from their base classes, but they do call the constructor and destructor of base classes.
- The default destructor calls the destructors of the base class and members of the derived class.
- The destructors of base classes and members are called in the reverse order of the completion of their constructor:
- The destructor for a class object is called before destructors for members and bases are called.

Copy Constructor

A copy constructor is a special constructor in the C++ programming language used to create a new object as a copy of an existing object.

A copy constructor is a constructor of the form **classname(classname &).**The compiler will use the copy constructors whenever you initialize an instance using values of another instance of the same type.

Copying of objects is achieved by the use of a copy constructor and assignment operator.

Example :

```
class Sample{ int i, j;}
public:
        Sample(int a, int b) // constructor
        { i=a;   j=b;      }
        Sample (Sample & s) //copy constructor
        {
                j=s.j ; i=s.j;
                Cout <<"\n Copy constructor working \n";
        }
        void print (void)
        {cout <<i<< j<< "\n";}
        :
};
```

Note : ***The argument to a copy constructor is passed by reference, the reason being that whenan argument is passed by value, a copy of it is constructed. But the copy constructor is creating a copy of the object for itself, thus, it calls itself. Again the called copy constructor requires another copy so again it is called.in fact it calls itself again and again until the compiler runs out of the memory .so, in the copy constructor, the argument must be passed by reference.***

The following cases may result in a call to a copy constructor:

• When an object is passed by value to a function:

The pass by value method requires a copy of the passed argument to be created for the function to operate upon .Thus to create the copy of the passed object, copy constructor is invoked

If a function with the following prototype:

void cpyfunc(Sample);   // Sample is a class

Then for the following function call

cpyfunc(obj1); // obj1 is an object of Sample type

The copy constructor would be invoked to create a copy of the obj1 object for use by cpyfunc().

• **When a function returns an object:**

When an object is returned by a function the copy constructor is invoked

Sample cpyfunc(); // Sample is a class and it is return type of cpyfunc()

If func cpyfunc() is called by the following statement

obj2 = cpyfunc();

Then the copy constructor would be invoked to create a copy of the value returned by cpyfunc() and its value would be assigned to obj2. The copy constructor creates a temporary object to hold the return value of a function returning an object.

1 & 2 Marks Solved Problems:

Q1 :- Answer the questions after going through the following class.

```
class Exam
{
        char Subject[20] ;
        int Marks ;
public :
        Exam() // Function 1
        {
                strcpy(Subject, “Computer” ) ; Marks = 0 ;}
                Exam(char P[ ]) // Function 2
        {
                strcpy(Subject, P) ;
                Marks=0 ;
        }
        Exam(int M) // Function 3
        {
                strcpy(Subject, “Computer”) ; Marks = M ;
        }
        Exam(char P[ ], int M) // Function 4
        {
                strcpy(Subject, P) ; Marks = M ;
        }
};
```

(a) Which feature of the Object Oriented Programming is demonstrated using Function 1, Function2, Function 3 and Function 4 in the above class Exam?

Ans:- Function Overloading (Constructor overloading)

(b) Write statements in C++ that would execute Function 3 and Function 4 of class Exam.

Ans:- Exam a(10); and Exam b(“Comp”, 10);

Q2 Consider the following declaration:

```
class welcome
{
public:
        welcome (int x, char ch); // constructor with parameter
        welcome(); // constructor without parameter
        void compute();
private:
        int x; char ch;
};
```

Which of the following are valid statements?

```
        welcome obj (33, ‘a9’);
        welcome obj1(50, ‘9’);
        welcome obj3();
        obj1= welcome (45, ‘T’);
        obj3= welcome;
```

Ans. Valid and invalid statements are

welcome obj (33, ‘a9’);	**valid**
welcome obj1(50, ‘9’);	**valid**
welcome obj3();	**invalid**
obj1= welcome (45, ‘T’);	**valid**
obj3= welcome;	**invalid**

2 Marks Practice Problems

Q1. What do you understand by constructor and destructor functions used in classes? How are these functions different from other member functions? 2

Q2. What do you understand by default constructor and copy constructor functions used in classes? How are these functions different from normal constructors? **2**

Q3 Given the following C++ code, answer the questions (i) & (ii). 2

```
class TestMeOut
{
public :
        ~TestMeOut()            // Function 1
        { cout << "Leaving the examination hall " << endl; }
        TestMeOut() // Function 2
        { cout << "Appearing for examination " << endl; }
        void MyWork() // Function 3
        { cout << "Attempting Questions " << endl; }
};
```

(*i*) In Object Oriented Programming, what is Function 1 referred as and when does it get invoked / called?

(*ii*) In Object Oriented Programming, what is Function 2 referred as and when does it get invoked / called?

INHERITANCE

- **Inheritance is the process by which new classes called *derived* classes are created from existing classes called *base* classes**.
- The derived classes have all the features of the base class and the programmer can choose to add new features specific to the newly created derived class.
- The idea of inheritance implements the **IS-A** relationship. For example, mammal IS-A animal, dog IS-A mammal hence dog IS-A animal as well and so on.

EXAMPLE OF SINGLE INHERITANCE

Consider a base class **Shape** and its derived class **Rectangle** as follow:

```
// Base class
class Shape
{
public:
        void setWidth(int w)
        {
                width = w;
        }
        void setHeight(int h)
        {
                height = h;
        }
protected:
        int width;
        int height;
};

// Derived class
class Rectangle: public Shape
{
public:
        int getArea()
        {
                return (width * height);
        }
};

int main(void)
{
        Rectangle Rect;
        Rect.setWidth(5);
        Rect.setHeight(7);
        // Print the area of the object.
        cout << "Total area: " << Rect.getArea() << endl;
        return 0;
}
```

When the above code is compiled and executed, it produces following result:

Total area: 35

Access Control and Inheritance:

A derived class can access all the non-private members of its base class. Thus base-class members that should not be accessible to the member functions of derived classes should be declared private in the base class. We can summarize the different access types according to who can access them in the following way:

Access	**public**	**protected**	**private**
Same class	yes	yes	yes
Derived classes	yes	yes	no
Outside classes	yes	no	no

A derived class inherits all base class methods with the following exceptions:

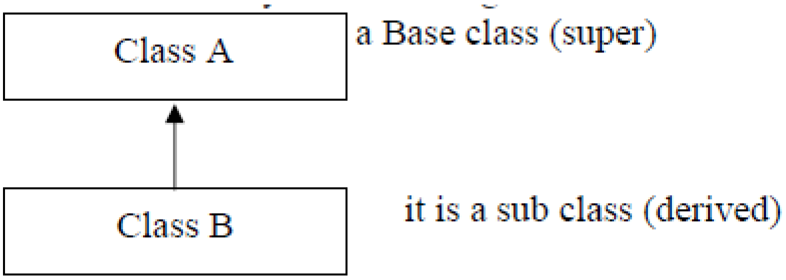
- Constructors, destructors and copy constructors of the base class.
- Overloaded operators of the base class.
- The friend functions of the base class.

When deriving a class from a base class, the base class may be inherited through **public, protected** or **private** inheritance. We hardly use **protected** or **private** inheritance but **public** inheritance is commonly used. While using different type of inheritance, following rules are applied:

1. **Public Inheritance:** When deriving a class from a **public** base class:
   - **public** members of the base class become **public** members of the derived class and
   - **protected** members of the base class become **protected** members of the derived class.
   - A base class's **private** members are never accessible directly from a derived class, but can be accessed through calls to the **public** and **protected** members of the base class.
2. **Protected Inheritance:** When deriving from a **protected** base class, **public** and **protected** members of the base class become **protected** members of the derived class.
3. **Private Inheritance:** When deriving from a **private** base class, **public** and **protected** members of the base class become **private** members of the derived Class.

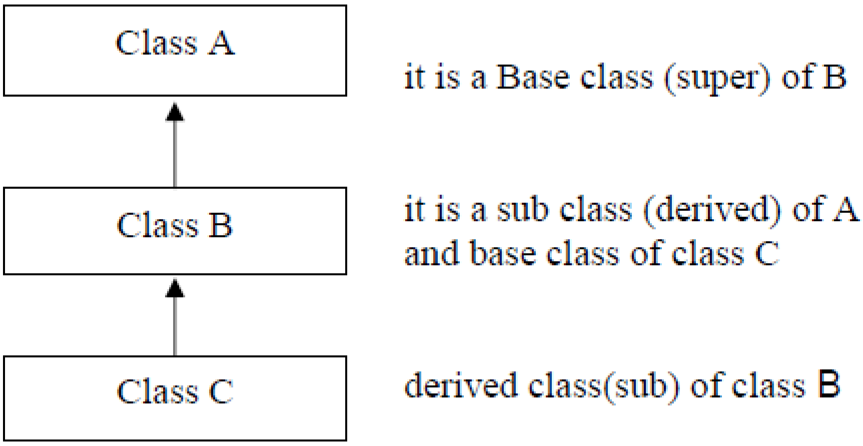
TYPES OF INHERITANCE

1. **Single class Inheritance:**
   - Single inheritance is the one where you have a single base class and a single derived class.

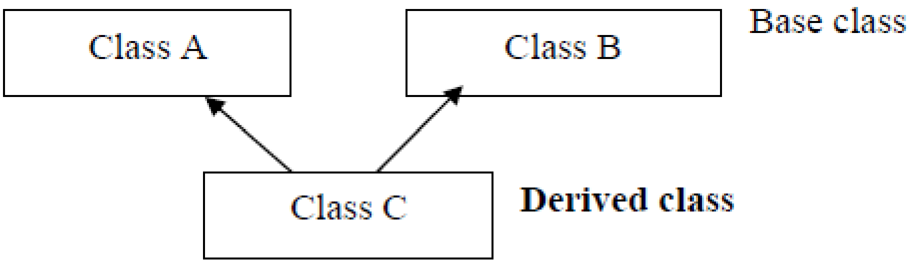


2. **Multilevel Inheritance:**

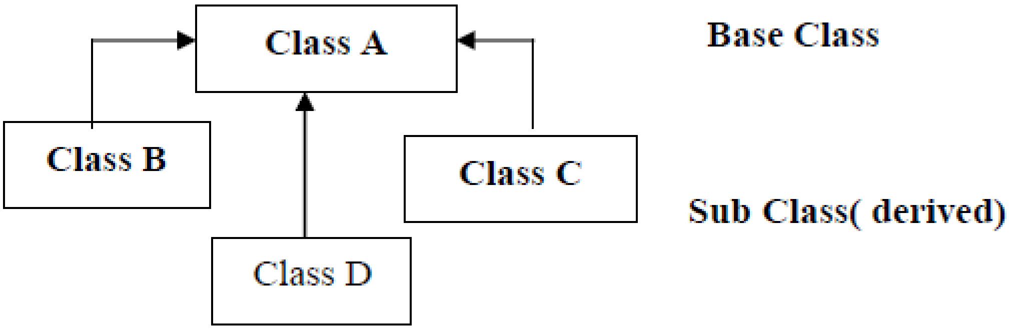
- In Multi level inheritance, a subclass inherits from a class that itself inherits from another class.


Class A
it is a Base class (super) of B
Class B
it is a sub class (derived) of A
and base class of class C
Class C
derived class(sub) of class B

3. **Multiple Inheritance:**
   - In Multiple inheritances, a derived class inherits from multiple base classes. It has properties of both the base classes.



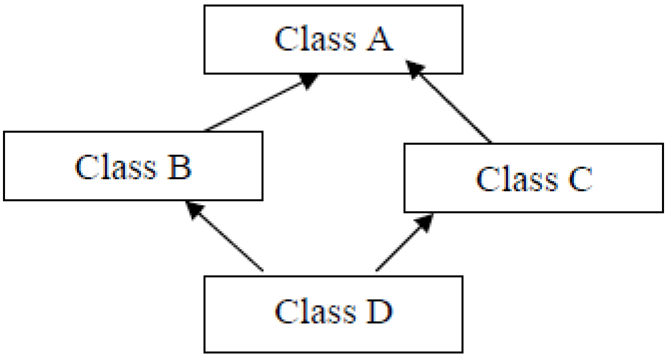
4. **Hierarchical Inheritance:**
   - In hierarchial Inheritance, it's like an inverted tree. So multiple classes inherit from a single base class.



5. **Hybrid Inheritance:**

- It combines two or more forms of inheritance .In this type of inheritance, we can have mixture of number of inheritances but this can generate an error of using same name function from no of classes, which will bother the compiler to how to use the functions.
- Therefore, it will generate errors in the program. This has known as ambiguity or duplicity.

- Ambiguity problem can be solved by using **virtual base classes**



4 marks Solved Problems:

Q1. **Consider the following declarations and answer the questions given below :**

```
class WORLD
{
        int H;
protected :
        int S;
public :
        void INPUT(int);
        void OUTPUT();
};
class COUNTRY : private WORLD  // Base class WORLD & Sub class Contry
{
        int T;
protected :
        int U;
public :
        void INDATA( int, int)
        void OUTDATA();
};
class STATE : public COUNTRY    // Base Class COUNTRY & Sub Class STATE
{
        int M;
public :
        void DISPLAY (void);
 };
```

(i) Name the base class and derived class of the class COUNTRY.

(ii) Name the data member(s) that can be accessed from function DISPLAY().

(iii) Name the member function(s), which can be accessed from the objects of class STATE.

(iv) Is the member function OUTPUT() accessible by the objects of the class COUNTRY ?

Ans:-

(i) Base class : WORLD
Derived class : STATE

(ii) M.

(iii)DISPLAY(), INDATA() and OUTDATA()

(iv)No

Q2. Consider the following declarations and answer the questions given below :

```
class living_being
{
        char name[20];
protected:
        int jaws;
public:
        void inputdata(char, int);
        void outputdata();
}
class animal : protected living_being
{
        int tail;
protected:
        int legs;
public:
        void readdata(int, int);
        void writedata();
};
class cow : private animal
{       char horn_size;
public:
        void fetchdata(char);
        void displaydata();
};
```

(i) Name the base class and derived class of the class animal.
(ii) Name the data member(s) that can be accessed from function displaydata.
(iii)Name the data member(s) that can be accessed by an object of cow class.
(iv)Is the member function outputdata accessible to the objects of animal class.

Ans:-

(i) Base class : living_being
Derived class : cow
(ii) horn_size, legs, jaws
(iii)etchdata() and displaydata()
(iv)No

Q3. **Consider the following and answer the questions given below:**

```
class MNC
{
        char Cname[25]; // Company name
protected :
        char Hoffice[25]; // Head office
public :
        MNC( );
        char Country[25];
        void EnterDate( );
        void DisplayData( );
};
class Branch : public MNC
{
        long NOE; // Number of employees
        char Ctry[25]; // Country
protected:
        void Association( );
public :
        Branch( );
        void Add( );
        void Show( );
};
class Outlet : public Branch
{
        char State[25];
public :
        Outlet();
        void Enter();
        void Output();
};
```

(i) Which class's constructor will be called first at the time of declaration of an object of class Outlet?

(ii) How many bytes an object belonging to class Outlet require ?

(iii)Name the member function(s), which are accessed from the object(s) of class Outlet.

(iv)**Name the data member(s), which are accessible from the object(s) of class Branch.**

Ans:-

(i) class MNC

(ii) 129

(iii)void Enter(), void Output(), void Add(), void Show(), void EnterData(), void DisplayData().

(iv)char country[25]

Q4 Consider the following and answer the questions given below :

```
class CEO
{
        double Turnover;
protected :
        int Noofcomp;
public :
        CEO( );
        void INPUT( );
        void OUTPUT( );
};
class Director : public CEO
{
        int Noofemp;
public :
        Director( );
        void INDATA();
        void OUTDATA( );
protected:
        float Funda;
};
class Manager : public Director
{
        float Expense;
public :
        Manager();
        void DISPLAY(void);
};
```

(i) Which constructor will be called first at the time of declaration of an object of class Manager?

(ii) How many bytes will an object belonging to class Manager require ?

(iii)Name the member function(s), which are directly accessible from the object(s) of class Manager.

(iv)**Is the member function OUTPUT() accessible by the objects of the class Director ?**

Ans:-

(i) CEO()

(ii) 16

(iii)DISPLAY(), INDATA(), OUTDATA(), INPUT(), OUTPUT()

(iv)Yes

4 marks Practice Problems:

Q1 :- Consider the following declarations and answer the questions given below:

```
class vehicle
{
    int wheels;
protected:
    int passenger;
public:
    void inputdata( int, int);
    void outputdata();
};

class heavyvehicle : protected vehicle
{
    int dieselpetrol;
protected:
    int load;
public:
    void readdata( int, int);
    void writedata();
};
class bus:private heavyvehicle
{
    char marks[20];
public:
    void fetchdata(char);
    void displaydata();
};
```

(i) Name the class and derived class of the class heavyvehicle.

(ii) Name the data members that can be accessed from function displaydata()

(iii)Name the data members that can be accessed by an object of bus class

(iv)Is the member function outputdata() accessible to the objects of heavyvehicle class.

Q2:- Consider the following declarations and answer the questions given below:

```
class book
{
        char title[20];
        char author[20];
        int noof pages;
public:
          void read();
          void show();
};
class textbook: private textbook
{
          int noofchapters, noofassignments;
protected:
          int standard;
        void readtextbook();
        void showtextbook();
};
class physicsbook: public textbook
{
          char topic[20];
public:
        void readphysicsbook();
        void showphysicsbook();
  };
```

(i) Name the members, which can be accessed from the member functions of class physicsbook.

(ii) Name the members, which can be accessed by an object of Class textbook.

(iii)Name the members, which can be accessed by an object of Class physicsbook.

(iv)What will be the size of an object (in bytes) of class physicsbook.

Q3 : Answer the questions (i) to (iv) based on the following:

```
class CUSTOMER
{
          int Cust_no;
          char Cust_Name[20];
protected:
          void Register();
public:
          CUSTOMER( );
```

```
                void Status( );
        };
        class SALESMAN
        {
                int Salesman_no;
                char Salesman_Name[20];
        protected:
                float Salary;
        public:
                SALESMAN( );
                void Enter( );
                void Show( );
        };
        class SHOP : private CUSTOMER, public SALESMAN
        {
                char Voucher_No[10];
                char Sales_Date[8;
        public :
                SHOP( );
                void Sales_Entry( );
                void Sales_Detail( );
          };
```

(i) Write the names of data members, which are accessible from object belonging to class CUSTOMER.

(ii) Write the names of all the member functions which are accessible from object belonging to class SALESMAN.

(iii)Write the names of all the members which are accessible from member functions of class SHOP.

(iv)How many bytes will be required by an object belonging to class SHOP?

2marks Practice Problems:

1. What is access specifier ? What is its role ?
2. What are the types of inheritance ?
3. What is the significance of inheritance ?
4. What is the difference between private and public visibility modes?

DATA FILE HANDLING IN C++

File:- A file is a stream of bytes stored on some secondary storage devices.

- **Text file:** A text file stores information in readable and printable form. Each line of text is terminated with an **EOL** (End of Line) character.
- **Binary file:** A binary file contains information in the non-readable form i.e. in the same format in which it is held in memory.

File Stream

- **Stream:** A stream is a general term used to name flow of data. Different streams are used to represent different kinds of data flow.

  There are three file I/O classes used for file read / write operations.

  - **ifstream -** can be used for read operations.
  - **ofstream -** can be used for write operations.
  - **fstream -** can be used for both read & write operations.
- **fstream.h**:-This header file includes the definitions for the stream classes ifstream, ofstream and fstream. In C++ **file input output** facilities implemented through **fstream.h** header file.
  It contain predefines set of operation for handling file related input and output, fstream class ties a file to the program for input and output operation.

A file can be opened using:

- **By the constructor method**. This will use default streams for file input or output. This method is preferred when file is opened in input or output mode only.
  Example : **ofstream file ("student.dat"); or ifstream file("student.dat");**

- **By the open() member function** of the stream. It will preferred when file is opened in various modes i.e **ios::in, ios::out, ios::app, ios::ate** etc.
  e.g **fstream file;**
  file.open("book.dat", ios::in | ios::out | ios::binary);

File modes:

Open Mode	**Description**
ios::out	It open file in output mode (i.e write mode) and place the file pointer in beginning, if file already exist it will overwrite the file.
ios::in	It open file in input mode (read mode) and permit reading from the file.
ios::app	It opens the file in write mode, and place file pointer at the end of file
ios::ate	It open the file in write or read mode, and place file pointer at the end of file
ios::trunc	It truncates the existing file (empties the file).
ios::nocreate	If file does not exist this file mode ensures that no file is created and open() fails.
ios::noreplace	If file does not exist, a new file gets created but if the file already exists, the open() fails.
ios::binary	Opens a file in binary mode

eof( ): This function determines the end-of-file by returning true(non-zero) for end of file otherwise returning false(zero).

close(): This function terminates the connection between the file and stream associated with it.

Stream_object.close();
e.g file.close();

Text File functions:

Char I/O :

- **get() –** read a single character from text file and store in a buffer. e.g **file.get(ch);**

- **put() -** writing a single character in textfile e.g. **file.put(ch);**
- **getline() -** read a line of text from text file store in a buffer. e.g **file.getline(s,80);**

- We can also use **file>>ch** for reading and **file<<ch** writing in text file.

Binary file functions:

- **read()-** read a block of binary data or reads a fixed number of bytes from the specified stream and store in a buffer.

  Syntax : Stream_object.read ((char *)& Object, sizeof (Object)) ;

  e.g file.read ((char *)&s, sizeof(s)) ;
- **write()** – write a block of binary data or writes fixed number of bytes from a specific memory location to the specified stream.

  Syntax : Stream_object.write((char *)& Object, sizeof(Object));

  e.g file.write((char *)&s, sizeof(s));

Note: Both functions take two arguments, **Initial address** and **length of the variable**

File Pointer: The file pointer indicates the position in the file at which the next input/output is to occur. There **read pointer** and **write pointer** associated with a file.

Moving the file pointer in a file for various operations viz modification, deletion , searching etc. Following functions are used:

- **seekg():** It places the file pointer to the specified position in input mode of file.

  e.g file.seekg(p,ios::beg); or file.seekg(-p,ios::end), or file.seekg(p,ios::cur)

  i.e to move to **p** byte position from beginning, end or current position.

- **seekp():** It places the file pointer to the specified position in output mode of file.

  e.g file.seekp(p,ios::beg); or file.seekp(-p,ios::end), or file.seekp(p,ios::cur)

  i.e to move to **p** byte position from beginning, end or current position.

- **tellg():** This function returns the current working position of the file pointer in the input mode.

  e.g int p=file.tellg( );

- **tellp():** This function returns the current working position of the file pointer in the output mode.

  e.f int p=file.tellp( );

Steps To Create A File

- ➢ Declare an object of the desired file stream class(ifstream, ofstream, or fstream)
- ➢ Open the required file to be processed using constructor or open function.
- ➢ Process the file.
- ➢ Close the file stream using the object of file stream.

General program structure used for creating a Text File

To create a text file using strings I/O

```
#include<fstream.h> //header file for file operations
                void main()
                {
                    char s[80], ch;
                    ofstream file(“myfile.txt”); //open myfile.txt in default output mode
                    do
                    {       cout<<”\n enter line of text”;
                            gets(s);        //standard input
```

```
                        file<<s;            // write in a file myfile.txt
                        cout<<"\n more input y/n";
                        cin>>ch;
                }while(  ch != 'n' || ch != 'N' ) ;
        file.close();
        } //end of main
```

To create a text file using characters I/O

```
        #include<fstream.h> //header file for file operations
        void main()
        {
        char ch;
        ofstream file("myfile.txt"); //open myfile.txt in default output mode
        do{
        ch=getche();
        if (ch==13) //check if character is enter key
        cout<<'\n';
        else
        file<<ch; // write a character in text file 'myfile.txt '
        } while(ch!=27); // check for escape key
        file.close();
        } //end of main
```

Text files in input mode:

To read content of 'myfile.txt' and display it on monitor.

```
        #include<fstream.h> //header file for file operations
        void main()
        {
                char ch;
                ifstream file("myfile.txt"); //open myfile.txt in default input mode
                while(file)
                {
                        file.get(ch) // read a character from text file ' myfile.txt'
                        cout<<ch; // write a character in text file 'myfile.txt '
                }
                file.close();
        } //end of main
```

2 Marks Questions:

1. **Write a function in a C++ to read the content of a text file "DELHI.TXT" and display all those lines on screen, which are either starting with 'D' or starting with 'M'. [CBSE 2012]**

```
        void DispDorM()
        {
                ifstream File("DELHI.TXT")
                char str[80];
                while(File.getline(str,80))
                {
                        if(str[0] = ='D' || str[0] = ='M')
                                cout<<str<<endl;
                }
                File.close(); //Ignore
        }
```

2. **Write a function in a C++ to count the number of lowercase alphabets present in a text file “BOOK.txt”.**

```
        int countalpha()
        {       ifstream Fin(“BOOK.txt”);
                char ch;
                int count=0;
                while(!Fin.eof())
                {
                        Fin.get(ch);
                        if (islower(ch))
                                count++;
                }
                Fin.close();
                return count;
        }
```

3. **Function to calculate the average word size of a text file.**

```
        void calculate()
        {
                fstream File;
                File.open(“book.txt”,ios::in);
                char a[20];
                char ch;
                int i=0,sum=0,n=0;
                while(File)
                {       File.get(ch);
                        a[i]=ch;
                        i++;
                        if((ch==’ ‘) || ch(== ‘.’)||(char==’,’)(ch==’\t’)||(ch==’\n’)
                        {
                                i --;
                                sum=sum +i;
                                i=0; N++;
                        }
                }
                cout<<”average word size is “<<(sum/n);
        }
```

4. **Assume a text file “coordinate.txt” is already created. Using this file create a C++ function to count the number of words having first character capital.**

```
        int countword()
        {       ifstream Fin(“BOOK.txt”);
                char ch[25];
                int count=0;
                while(!Fin.eof())
```

```
    {
        Fin>>ch;
        if (isupper(ch[0]))
           count++;
    }
    Fin.close();
    return count;
}
```

5. **Function to count number of lines from a text files (a line can have maximum 70 characters or ends at ‘.’)**

```
int countword()
{
    ifstream Fin(“BOOK.txt”);
    char ch[70];
    int count=0;
    if (!Fin)
    {
        cout<<”Error opening file!” ;
        exit(0);
    }
    while(1)
    {
        Fin.getline(ch,70,‘.’);
        if (Fin.eof())
            break;
        count++;
    }
    Fin.close();
    return count;
}
```

2/3 Marks Practice Questions

1. Write a function in C++ to count the number of uppercase alphabets present in a text file “BOOK.txt”
2. Write a function in C++ to count the number of alphabets present in a text file “BOOK.txt”
3. Write a function in C++ to count the number of digits present in a text file “BOOK.txt”
4. Write a function in C++ to count the number of white spaces present in a text file “BOOK.txt”
5. Write a function in C++ to count the number of vowels present in a text file “BOOK.txt”
6. Assume a text file “Test.txt” is already created. Using this file, write a function to create three files “LOWER.TXT” which contains all the lowercase vowels and “UPPER.TXT” which contains all the uppercase vowels and “DIGIT.TXT” which contains all digits.

General program structure used for operating a Binary File

1. **Program to create a binary file ‘student.dat’ using structure.**

```
#include<fstream.h>
struct student
{
```

```
            char name[15];
            float percent;
        };
        void main()
        {
            ofstream fout;  // f out is output file stream it will open file in write mode
            char ch;
            fout.open(“student.dat”, ios::out | ios:: binary);      // student.dat will be opened in
                                                                    // binary Mode
            clrscr();
             student s;      //s is a variable of type student that contains name & percent
            do
            {       // inputting data to record  s
                  cout<<”\n enter name of student”;
                  gets(s);
                  cout<<”\n enter persentage”;
                  cin>>percent;
                  //Writing contents of s to file student.dat
                  fout.write ( ( char * ) &s, sizeof ( s ) ) ;
                  cout<<”\n more record y/n”;
                  cin>>ch;
            }while(ch!=’n’ || ch!=’N’) ;
            fout.close();
      }
```

2. **Program to read a binary file ‘student.dat’ display records on monitor.**

```
      #include<fstream.h>
      struct student
      {
            char name[15];
            float percent;
      };
      void main()
      {
            ifstream fin;    //fin is an input file stream that can open a file in read mode
            student s;        // s is a record of type student
            fin.open(“student.dat”,ios::in | ios:: binary); // opening student.dat in binary mode
            fin.read((char *) &s, sizeof(student)); //read a record from file ‘student.dat’
            invariable s
            while(file) // file will read until end of file does not come.
            {
                  //Displaying the content of  s (reord) read from the student.dat
                  cout<<s.name;
                  cout<<“\n has the percent: ”<<s.percent;
                  fin.read((char *) &s, sizeof(student)); // reading the next record
             }
            fin.close();
      }
```

Binary file using Objects and other file operations:

1. **Consider the following class declaration then write c++ function for following file operations viz create_file, read_file, add new records, modify record, delete a record, search for a record.**

```
#include<iostream.h>
class student
{
        int rno;
        char name[30];
        int age;
public:
        void input( )    // function to input values for data member of current object
        {
                cout<<”\n enter roll no”;
                cin>>rno;
                cout<<”\n enter name “;
                gets(name);
                cout<<”\n enter age”;
                cin>>age;
        }
        void output( )             // function to display contents of current object
        {
                cout<< “\n roll no:”<<rno;
                cout<< “\n name :”<<name;
                cout<< “\n age:”<<age;
        }
        int getrno( )              // function to get  the rno from current object
        {
                return rno;
        }
};

void create_file( )       // function to create a blank file student.dat
{
        ofstream fout;  // fout is output file stream object that will open a file in write
mode
        fout.open(“student”, ios::out | ios:: binary); // opening file in binary mode
        fout.close();     // closing file student.dat
}

void read_file( )            //function to read records from student.dat one by into record s
{
        ifstream fin;
        student s;
        fin.open(“student.dat”,ios::in | ios:: binary); //opening the file
        fin.read((char *) &s,sizeof(student));      //reading first record from file to record s
        while(file)
        {
                s.output();         // displaying the content of object s (record)
                cout<< “\n”;
                fin.read ( ( char * ) & s,sizeof ( student ) ) ; // reading next record
        }
        fin.close();
}

void modify_record()
{
        student s;
        fstream file;
        file.open(“student.dat”,ios::in|ios::out|ios::ate|ios::binary); // opening student.dat in read
        &
                                                                        //write binary mode
        int r, pos = -1, f=0;
        cout<<”\n enter the rollo no of student whom data to be modified”;
        cin>>r ;
        //Searching the record with rno = r
        file.read( ( char * )&s, sizeof( s ) ) ;
        while(file)
        {
                if (r == s.getrno( )) // will be true if required record found (rno = r)
                {
                        f=1;      // f = 1 indicate record found
                        cout<<”\n record is “;
                        s.output();
                        pos =file.tellg()-size(s); //moving the write pointer one record back
                        break;
                }
                file.read((char *)&s,sizeof(s));   // writing the modified record to the file
        }
        if(f == 0 )          // f==0 indicate record did not found
                cout<< “\n record not exist”;
}
void delete_record()
{
        fstream file(“student.dat”, ios::in|ios::binary);
        fstream newfile(“newstu.dat”,ios::out|ios::binary);
        student s;
        cout<<”\n enter the rollno no of student whom record to be deleted”;
        cin>>r;
        file.read ( (char *)&s, sizeof( s ) ) ;
        while(file)
        {
                if (r!=s.getrno())
                {
                        newfile.write((char *)&s,sizeof(s));
                }
                file.read((char *)&s,sizeof(s));
        }
        file.close();
        newfile.close();
}
```

```
void search_record()
{           student s;
            fstream file;
            file.open("student.dat",ios::in|os::binary);
            int r,flag = 0;
            cout<<"\n enter the rollo no of student whom record to be searched";
            cin>>r;
            file.read( ( char * )&s, sizeof( s ) ) ;
            while(file)
            {
                        if (r==s.getrno())
                        {           cout<<"\n record is ";
                                    s.output();
                                    flag=1;
                                    break;
                        }
                        file.read((char *)&s,sizeof(s));
            }
            if(flag==0)
                        cout<< "\n search unsuccessfull";
            file.close();
}
```

1 Mark Questions

1. Observe the program segment carefully and answer the question that follows:

```
class stock
{
int Ino, Qty; Char Item[20];
public:
void Enter() { cin>>Ino; gets(Item); cin>>Qty;}
void issue(int Q) { Qty+=0;}
void Purchase(int Q) {Qty-=Q;}
int GetIno() { return Ino;}
};
void PurchaseItem(int Pino, int PQty)
{ fstream File;
47
File.open("stock.dat", ios::binary|ios::in|ios::out);
Stock s;
int success=0;
while(success= = 0 && File.read((char *)&s,sizeof(s)))
{
If(Pino= = ss.GetIno())
{
s.Purchase(PQty);
_________________________ // statement 1
_________________________ // statement 2
Success++;
}
}
if (success = =1)
cout<< “Purchase Updated”<<endl;
else
cout<< “Wrong Item No”<<endl;
File.close() ;
}
```

Ans.1.

i) Statement 1 to position the file pointer to the appropriate place so that the data updation is done for the required item.

```
File.seekp( File.tellg( ) - sizeof(stock);
                    OR
File.seekp(-sizeof(stock),ios::cur);
```

ii) Staement 2 to perform write operation so that the updation is done in the binary file.

```
File.write((char *)&s, sizeof(s));
                    OR
File.write((char *)&s, sizeof(stock));
```

3 Marks Question

1. **Write a function in c++ to search for details (Phoneno and Calls) of those Phones which have more than 800 calls from binary file “phones.dat”. Assuming that this binary file contains records/ objects of class Phone, which is defined below.**

class Phone **CBSE 2012**

```
{
Char Phoneno[10]; int Calls;
public:
void Get() {gets(Phoneno); cin>>Calls;}
void Billing() { cout<<Phoneno<< “#”<<Calls<<endl;}
int GetCalls() {return Calls;}
};
```

Ans 1 :

```
void Search()
{
Phone P;
fstream fin;
fin.open( “Phone.dat”, ios::binary| ios::in);
while(fin.read((char *)&P, sizeof(P)))
{
if(p.GetCalls() >800)
p.Billing();
}
Fin.close(); //ignore
}};
```

2. **Write a function in C++ to add new objects at the bottom of a binary file “STUDENT.DAT”, assuming the binary file is containing the objects of the following class.**

```
class STUD
{int Rno;
char Name[20];
public:
void Enter()
{cin>>Rno;gets(Name);}
void Display(){cout<<Rno<<Name<<endl;}
};
```

Ans.2.

```
void searchbook(int bookno)
{ifstream ifile(“BOOK.DAT”,ios::in|ios::binary);
if(!ifile)
{cout<<”could not open BOOK.DAT file”; exit(-1);}
else
{BOOK b; int found=0;
while(ifile.read((char *)&b, sizeof(b)))
{if(b.RBno()==bookno)
{b.Display(); found=1; break;}
}
if(! found)
cout<<”record is not found “;
ifile.close();
}
}
```

3. **Given a binary file PHONE.DAT, containing records of the following class type**

```
class Phonlist
{
char name[20];
char address[30];
char areacode[5];
char Phoneno[15];
public:
void Register()
void Show();
void CheckCode(char AC[])
{return(strcmp(areacode,AC);
}};
```

Write a function TRANSFER( ) in C++, that would copy all those records which are having areacode as “DEL” from PHONE.DAT to PHONBACK.DAT.

Ans 3.

```
void TRANSFER()
{
fstream File1,File2;
Phonelist P;
File1.open(“PHONE.DAT”, ios::binary|ios::in);
File2.open(“PHONEBACK.DAT”, ios::binary|ios::OUT)
while(File1.read((char *)&P, sizeof(P)))
{ if( p.CheckCode( “DEL”))
File2.write((char *)&P,sizeof(P)); }
File1.close();
File2.close();
}
```

POINTERS

Pointer is a variable that holds a memory address of another variable of same type.
- It supports dynamic allocation routines.

C++MemoryMap :

- Program Code : It holds the compiled code of the program.
- Global Variables : They remain in the memory as long as program continues.
- Stack : It is used for holding return addresses at function calls, arguments passed to the functions, local variables for functions. It also stores the current state of the CPU.
- Heap : It is a region of free memory from which chunks of memory are allocated via DMA functions.

StaticMemory Allocation : Allocation of memory at compile time.

```
e.g. int a;       // This will allocate 2 bytes for a during compilation.
```

Dynamic Memory Allocation : allocation of memory at run time using operator **new and delete**.

```
e.g int x =new int;     // dynamic allocation
float y= new float;
delete x;               //dynamic deallocation
delete y;
```

Free Store : It is a pool of unallocated heap memory given to a program that is used by the program for dynamic memory allocation during execution.

Declaration and Initialization of Pointers:

Syntax : Datatype *variable_name;

```
int *p;       // p is an integer pointer contains address of an integer variable
float *p1;    // p1 is a float pointer contains address of a floating point variable
char *c;      // c is a character pointer contains address of a character variable

int a = 10;         // a is an integer variable hold value 10
int *p = &a;        //pointer initialization [ p is a integer pointer  holds address of a]
                    // &a means address of a
```

Pointer arithmetic:

Two arithmetic operations, addition and subtraction, may be performed on pointers.
Adding 1 to a pointer actually adds the size of pointer's base type.
Base address: The address of the first byte is known as BASE ADDRESS.

Dynamic Allocation Operators:

int * p = new int[10]; this will dynamically allocate 20 bytes (10*2) to integer pointer p

Two dimensional arrays:

```
int *arr, r, c;
r = 5; c = 5;
arr = new int [r * c];
```

Now to read the element of array, you can use the following loops :

```
For (int i = 0; i < r; i++)
{
        cout << “\n Enter element in row “ << i + 1 << “ : “;
        For (int j=0; j < c; j++)
                cin >> arr [ i * c + j];
}
```

Memory released with delete as below:

```
delete arr;
```

Pointers and Arrays:

Array of Pointers:

To declare an array holding 5 int pointers –

```
int * ip[5];
```

That would be allocated for 5 pointers that can point to integers.

```
ip[0] = &a;    ip[1] = &b;    ip[2] = &c;    ip[3] = &d;    ip[4] = &e;
```

I p	Address of a	Address of b	Address of c	Address of d	Address of e

Pointers and Strings:

Pointer is very useful to handle the character array also. E.g :

```
    void main()
    {
            char str[ ] = “computer”;
            char *cp;
            cp = str;           // cp will hold address of first element of array str
            cout<<str ;         //display string
            cout<<cp;           // display string
            for (cp =str; *cp != ‘\0’; cp++) // display character by character by character
            cout << ”--“<<*cp;
    }
```

Output :

```
Computer
Computer
--c--o--m--p--u--t--e—r
```

Pointers and CONST :

A **constant pointer** means that the pointer in consideration will always point to the same address. Its address cannot be modified.

Int n = 20;

int * const c = &n; // a constant pointer c to an integer n

A **pointer to a constant** refers to a pointer which is pointing to a symbolic constant.

const int b = 10;   // a constant integer b

**const int ***pc = &b;   // a pointer to a constant integer b

Pointers and Functions :

Invoking Function by Passing the Pointers:

When the pointers are passed to the function, the addresses of actual arguments in the calling function are copied into formal arguments of the called function.

```
#include<iostream.h>
void swap(int *m, int *n)
{
        int temp;
        temp = *m;
        *m = *n;
        *n = temp;
}

void main()
{
        void swap(int *m, int *n);
        int a = 5, b = 6;
        cout << "\n Value of a :" << a << " and b :" << b;
        swap(&a, &b);
        cout << "\n After swapping value of a :" << a << "and b :" << b;
}
```

```
Input :
Value of a : 5 and b : 6
After swapping value of a : 6 and b : 5
```

Function returning Pointers :

A function can also returns a pointer.

Syntax:-   type * function-name (argument list);

```
#include <iostream.h>
int *min(int  &x,  int  &y)      // function returning int pointer
{
        if (x < y )
                return (&x);
        else
                return (&y)
}
void main()
{
        int a, b, *c;
        cout << "\nEnter a :"; cin >> a;
        cout << "\nEnter b :"; cint >> b;
        c = min(a, b);  // pointer returned from the function min will be assigned to pointer variable c
        cout << "\n The minimum no is :" << *c;
}
```

Dynamic structures:

- **Dynamic allocation:-**

  The new operator can be used to create dynamic structures also.

  ```
  struct  student
  {
          int rno;
          char name[20];
  };
  student *s = new student;
  ```

  To access structure member through structure pointer we use **arrow operator**(->).

  ```
  cout << s -> rno;
  cout << s -> name
  ```

- **Dynamic deallocation:-**

  A dynamic structure can be released using the deallocation operator **delete** as shown below :

  ```
  delete stu;
  ```

this Pointer : refers to the address of current object.

Solved Questions

Q1. How is *p different from **p ?

Ans : *p means, it is a pointer pointing to a memory location storing a value in it. But **p means, it is a pointer pointing to another pointer which in turn points to a memory location storing a value in it.

Q2. How is &p different from *p ?

Ans : &p gives us the address of variable p and *p. dereferences p and gives us the value stored in memory location pointed to by p.

Q3. Find the error in following code segment :

```
Float **p1, p2;
P2 = &p1;
```

Ans : In code segment, p1 is pointer to pointer, it means it can store the address of another pointer variable, whereas p2 is a simple pointer that can store the address of a normal variable. So here the statement p2 = &p1 has error.

Q. 4 What will be the output of the following code segment ?

```
char C1 = ‘A’;
char C2 = ‘D’;
char *i, *j;
i = &C1;
j = &C2;
*i = j;
cout << C1;
```

Ans : It will print A.

Q. 5 How does C++ organize memory when a program is run ?

Ans : Once a program is compiled, C++ creates four logically distinct regions of memory :

- area to hold the compiled program code
- area to hold global variables

- the stack area to hold the return addresses of function calls, arguments passed to the functions, local variables for functions, and the current state of the CPU.
- The heap area from which the memory is dynamically allocated to the program.

Q. 6 Identify and explain the error(s) in the following code segment :

```
float a[] = { 11.02, 12.13, 19.11, 17.41};
float *j, *k;
j = a;
k = a + 4;
j = j * 2;
k = k / 2;
cout << " *j = " << *j << ", *k = " << *k << "\n";
```

Ans : The erroneous statements in the code are :

```
j = j * 2;
k = k / 2;
```

Because multiplication and division operations cannot be performed on pointer and j and k are pointers.

Q7. How does the functioning of a function differ when

(i) an object is passed by value ?

(ii) an object is passed by reference ?

Ans :

(i) When an object is passed by value, the called function creates its own copy of theobject by just copying the contents of the passed object. It invokes the object's copy constructor to create its copy of the object. However, the called function destroys its copy of the object by calling the destructor function of the object upon its termination.

(ii) When an object is passed by reference, the called function does not create its own copy of the passed object. Rather it refers to the original object using its reference or alias name. Therefore, neither constructor nor destructor function of the object is invoked in such a case.

2 MARKS PRACTICE QUESTIONS

1. Differentiate between static and dynamic allocation of memory.
2. Identify and explain the error in the following program :

```
#include<iostream.h>
int main()
{int x[] = { 1, 2, 3, 4, 5 };
for (int i = 0; i < 5; i++)
{
        cout << *x;
        x++;
}r
eturn 0;
}
```

3. Give the output of the following :

```
char *s = "computer";
for (int x = strlen(s) – 1; x >= 0; x--)
```

```
{
for(int y =0; y <= x; y++) cout << s[y];
cout << endl;
}
```

4. Identify the syntax error(s), if any, in the following program. Also give reason for errors.

```
void main()
{const int i = 20;
const int * const ptr = &i;
(*ptr++; int j= 15; ptr
= &j; }
```

5. What is ‘this’ pointer? What is its significance?
6. What will be the output of following program ?

```
#include<iostream.h>
void main()
{
char name1[] = “ankur”; char
name2[] = “ankur”; if (name1 !=
name2)
cout << “\n both the strings are not equal”;
else
cout << “\n the strings are equal”; }
```

7. Give and explain the output of the following code :

```
void junk (int, int *);
int main() {
int i = 6, j = -4;
junk (i, &j);
cout << “i = “ << i << “, j = “ << j << “\n”;
return 0; }
void junk(int a, int *b)
{
a = a* a;
*b = *b * *b; }
```

UNIT-2 DATA STRUCTURES

A **data structure** is a particular way of storing and organizing data in a computer so that it can be used efficiently. The data structure can be classified into following two types:

- **Simple Data Structure:** These data structures are normally built from primitive data types like integers,
  floats, characters. For example arrays and structure.
- **Compound Data Structure:** simple data structures can be combined in various ways to form more complex structure called compound structures. Linked Lists, Stack, Queues and Trees are examples of compound data structure.

Data Structure Arrays

Data structure array is defined as linear sequence of finite number of objects of same type with following set of operation:

- Creating : defining an array of required size
- Insertion: addition of a new data element in the in the array
- Deletion: removal of a data element from the array
- Searching: searching for the specified data from the array
- Traversing: processing all the data elements of the array
- Sorting : arranging data elements of the array in increasing or decreasing order
- Merging : combining elements of two similar types of arrays to form a new array of same type

In C++ an array can be defined as

Datatype   arrayname [size] ;

Where size defines the maximum number of elements can be hold in the array.

For example

```
 float b[10];     //b is an array which can store maximum 10 float values
int c[5];         //c is an array which can store maximum 5 integer values
```

Array initialization

```
void main()
{
        int b[10]={3,5,7,8,9};           // array initialization
        cout<<b[4]<<endl;
        cout<<b[5]<<endl;
}
```

Output is

```
9
0
```

Searching

We can use two different search algorithms for searching a specific data from an array

- Linear search algorithm
- Binary search algorithm

Linear search algorithm

In Linear search, each element of the array is compared with the given item to be searched for. This method continues until the searched item is found or the last item is compared.

```
#include<iostream.h>
int linear_search(int a[], int size, int item)
```

```
{
        int i=0;
        while( i<size && a[i] !=item)
                i++;
        if(i<size)
                return ( i);        //returns the index number of the item in the array
        else
                return (-1);        //given item is not present in the array so it returns -1
                                    //since -1 is not a legal index number
}
void main()
{
        int b[8]={2,4,5,7,8,9,12,15},size=8;
        int item;
        cout<<”enter a number to be searched for”;
        cin>>item;
        int p = linear_search(b, size, item);     //search item in the array b
        if(p == -1)
                cout<<item<<” is not present in the array”<<endl;
        else
                cout<<item <<” is present in the array at index no “<<p;
}
```

Binary search algorithm

Binary search algorithm is applicable for already sorted array only. In this algorithm, to search for the given item from the sorted array (in ascending order),

- The item is compared with the middle element of the array. If the middle element is equal to the item then index of the middle element is returned
- If item is less than the middle item then the item will be searched in first half segment of the array for the next iteration.
- If the item is larger than the middle element then the item will be searched in second half of the array for the next iteration
- The same process continues until either the item is found or the segment is reduced to the single element and still the item is not found (search unsuccessful).

```
#include<iostream.h>
int binary_search(int a[ ], int size, int item)
{
        int first = 0, last = size-1, middle;
        while(first<=last)
        {
                Middle = ( first + last ) / 2;
                if( item = = a[middle])
                        return middle;          // item is found
                else if(item< a[middle])
                        last=middle-1; //item is present in left side of the middle element
                else
                        first=middle+1; // item is present in right side of the middle element
        }
        return -1; //given item is not present in the array, here, -1 indicates unsuccessful search
}
void main()
{
         int b[8]={2,4,5,7,8,9,12,15},size=8;
        int item;
        cout<<"enter a number to be searched for";
        cin>>item;
        int p=binary_search(b, size, item); //search item in the array b
        if(p = = -1)
                  cout<<item<<" is not present in the array"<<endl;
         else
                  cout<<item <<" is present in the array at index no "<<p;
}
```

Steps to find item 12 in the array b

First	Last	middle	B[middle] = item	comment
Item 12 will be search in array b {2,4,5,7,9,12,15}				
0	7	7/2 = 3	b[3] = 7	Item not found , item 12 is greater than 7
Item 12 will be search in second segment array b {9,12,15}				
4	7	11/2 = 5	b[5] = 9	Item not found, 12 is greater than 9
Now Item will be searched 2nd half of previous segment b{12,15}				
6	7	13/2 = 6	B[6]=12	Item found & function will return 6 (position of item in array b
*if first > last, the function will return -1 (item not found)				

Output

12 is present in the array at index no 6

Inserting a new element in an array:-

We can insert a new element in an array in two ways

- If the array is unordered, the new element is inserted at the end of the array
- If the array is sorted then the new element is added at appropriate position without altering the order. To achieve this, all elements greater than the new element are shifted. For example, to add 10 in the given array below:

a[0]	a[1]	[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]
2	4	5	7	8	11	12	15	

Original array

a[0]	a[1]	[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]
2	4	5	7	8		11	12	15

Elements greater than 10 shifted to create free place to insert 10

a[0]	a[1]	[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]
2	4	5	7	8	10	11	12	15

Array after insertion

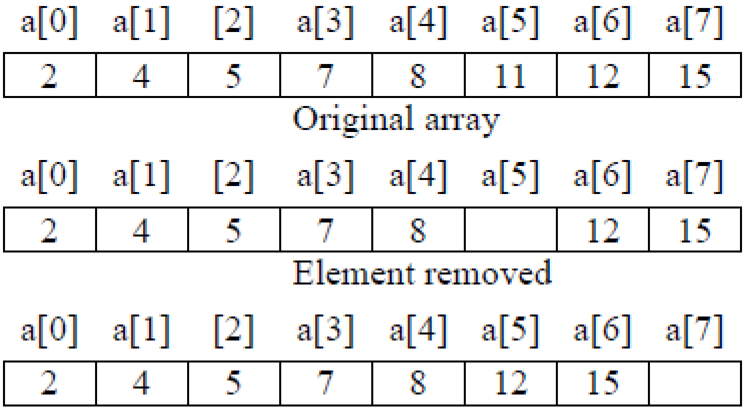
Following program implement insertion operation for sorted array

```
#include<iostream.h>
void insert(int a[ ], int &n, int item) //n is the number of elements already present in the array
{
        int i=n-1;
        while (i>=0 && a[i]>item)
        {
                a[i+1]=a[i]; // shift the ith element one position towards right
                i--;
        }
        a[i+1]=item; //insertion of item at appropriate place
        n++; //after insertion, number of elements present in the array is increased by 1
}
void main()
{
        int a[10]={2,4,5,7,8,11,12,15},n=8;
        int i=0;

        cout<<"Original array is:\n";  // displaying original array
        for(i=0;i<n;i++)
                cout<<a[i]<<", ";

        insert(a,n,10);                // inserting new element

        cout<<"\nArray after inserting 10 is:\n";      // displaying array after insertion
        for(i=0; i<n; i++)
                cout<<a[i]<<", ";
}
```

```
Output is
Original array is:
2, 4, 5, 7, 8, 11, 12, 15
Array after inserting 10 is:
2, 4, 5, 7, 8, 10, 11, 12, 15
```

Deletion of an item from a sorted array

In this algorithm the item to be deleted from the sorted array is searched and if the item is found in the array then the element is removed and the rest of the elements are shifted one position toward left in the array to keep the ordered array undisturbed. Deletion operation reduces the number of elements present in the array by1. For example, to remove 11 from the given array below:



a[0]	a[1]	[2]	a[3]	a[4]	a[5]	a[6]	a[7]
2	4	5	7	8	11	12	15

Original array

a[0]	a[1]	[2]	a[3]	a[4]	a[5]	a[6]	a[7]
2	4	5	7	8		12	15

Element removed

a[0]	a[1]	[2]	a[3]	a[4]	a[5]	a[6]	a[7]
2	4	5	7	8	12	15	

Following program implement deletion operation for sorted array

```
#include<iostream.h>
void delete_item(int a[ ], int &n, int item) //n is the number of elements already present in the array
{
        int i=0;
        while(i<n && a[i]<item)
                i++;
        if (a[i]==item) // given item is found
        {
                while (i<n)
                {
                        a[i]=a[i+1]; // shift the (i+1)th element one position towards left
                        i++;
                }
                cout<<"\n Given item is successfully deleted";
        }
        else
                cout<<"\n Given item is not found in the array";
        n--;
}
void main()
{
        int a[10]={2,4,5,7,8,11,12,15},n=8;
        int i=0;

        cout<<"Original array is :\n"; // displaying original arrary
        for(i=0;i<n;i++)
                cout<<a[i]<<", ";

        delete_item(a,n,11);            //deleting an item from array

        cout<<"\nArray after deleting 11 is:\n";        // displaying array after deletion
        for(i=0; i<n; i++)
                cout<<a[i]<<", ";
}
```

```
Output is
Original array is:
2, 4, 5, 7, 8, 11, 12, 15
Given item is successfully deleted
Array after deleting 11 is:
2, 4, 5, 7, 8, 12, 15
```

Traversal

Processing of all elements (i.e. from first element to the last element) present in one-dimensional array is called traversal. For example, printing all elements of an array, finding sum of all elements present in an array.

```
#include<iostream.h>
void print_array(int a[ ], int n) //n is the number of elements present in the array
{
        int i;
        cout<<"\n Given array is :\n";
        for(i=0; i<n; i++)
                cout<<a[i]<<", ";
}

int sum(int a[ ], int n)
{
        int i,s=0;
        for(i=0; i<n; i++)
                s=s+a[i];
        return s;
}

void main()
{
        int b[10]={3,5,6,2,8,4,1,12,25,13},n=10;
        int i, s;
        print_array(b,n);
        s = sum(b,n);
        cout<<"\n Sum of all elements of the given array is : "<<s;
}
```

```
Output is
Given array is
3, 5, 6, 2, 8, 4, 1, 12, 25, 13
Sum of all elements of the given array is : 79
```

Sorting

The process of arranging the array elements in increasing (ascending) or decreasing (descending) order is known as sorting. There are several sorting techniques are available e.g. selection sort, insertion sort, bubble sort, quick sort, heap short etc. But in CBSE syllabus only selection sort, insertion sort, bubble sort are specified.

Selection Sort

The basic idea of a selection sort is to repeatedly select the smallest element in the remaining unsorted array and exchange the selected smallest element with the first element of the unsorted array. For example, consider the following unsorted array to be sorted using selection sort

	A[0]	A[1]	A[2]	A[3]	A[4]
Original array	**10**	**2**	**20**	**5**	**9**
1st iteration	Find the smallest element & swap with 1st element				
Array after 1st iteration	2	**10**	**20**	**5**	**9**
2nd iteration		Find smallest element & swap with 2nd element			
Array after 2nd iteration	2	5	**20**	**10**	**9**
3rd iteration			Find smallest element & swap with 3rd element		
Array after 3rd iteration	2	5	9	**10**	**20**
4th iteration				Find smallest element & swap with 4th element	
Array after 4th iteration	2	5	9	10	**20**

- Normal numbers are sorted **& Bold numbers are unsorted list**

```
#include<iostream.h>
void select_sort(int a[ ], int n)          //n is the number of elements present in the array
{
        int i, j, p, small;
        for(i=0;i<n-1;i++)
        {
                small=a[i]; // initialize small with the first element of unsorted part of the array
                p=i; // keep index of the smallest number of unsorted part of the array in p
        for(j=i+1; j<n; j++) //loop for selecting the smallest element form unsorted array
        {
            if(a[j]<small)
            {
                small=a[j];
                p=j;
             }
         }// end of inner loop----------
         //----------exchange the smallest element with ith element-------------
         a[p]=a[i];
         a[i]=small;
        //-----------end of exchange-------------
    }
}       //end of function

void main( )
{
    int a[7] = {8,5,9,3,16,4,7}, n = 7, i;
    cout<<"\n Original array is :\n";
    for(i=0;i<n;i++)
        cout<<a[i]<<", ";

    select_sort(a,n);

    cout<<"\nThe sorted array is:\n";
    for(i=0; i<n; i++)
        cout<<a[i]<<", ";
}
```

Output is

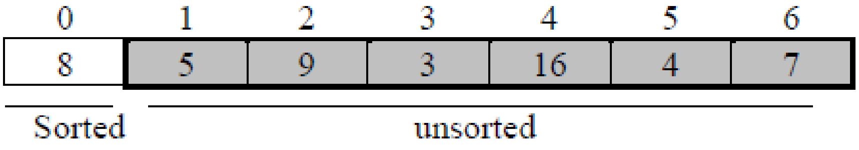
```
Original array is
8, 5, 9, 3, 16, 4, 7
The sorted array is
3, 4, 5, 7, 8, 9, 16
```

Insertion Sort

Insertion sort algorithm divides the array of n elements in to two subparts, the first subpart contain a[0] to a[k] elements in sorted order and the second subpart contain a[k+1] to a[n] which are to be sorted. The algorithm starts with only first element in the sorted subpart because array of one element is itself in sorted order. In each pass, the first element of the unsorted subpart is removed and is inserted at the appropriate position in the sorted array so that the sorted array remain in sorted order and hence in each pass the size of the sorted subpart is increased by 1 and size of unsorted subpart is decreased by 1. This process continues until all n-1 elements of the unsorted arrays are inserted at their appropriate position in the sorted array.

For example, consider the following unsorted array to be sorted using selection sort

Original array



0	1	2	3	4	5	6
8	5	9	3	16	4	7
Sorted	unsorted					

Original array	S[0]	**S[1]**	**S[2]**	**S[3]**	**S[4]**	**S[5]**	**S[6]**
	8	**5**	**9**	**3**	**16**	**4**	**7**
After 1st iteration	5	8	**9**	**3**	**16**	**4**	**7**
After 2nd iteration	5	8	9	**3**	**16**	**4**	**7**
After 3rd iteration	3	5	8	9	**16**	**4**	**7**
After 4th iteration	3	5	8	9	16	**4**	**7**
After 5th iteration	3	4	5	8	9	16	**7**
After 6th iteration	3	4	5	7	8	9	16

```
#include<iostream.h>
void insert_sort(int a[ ],int n) //n is the no of elements present in the array
{
        int i, j,p;
        for (i=1; i<n; i++)
        {
                p = a[i];
                j = i-1;
                //inner loop to shift all elements of sorted subpart one position towards right
                while( j >= 0 & &a [ j ] > p)
                {
                        a[j+1] = a[j];
                        j--;
                }               //---------end of inner loop
                a[j+1] = p; //insert p in the sorted subpart
        }
}
void main( )
{
        int a[7]={8,5,9,3,16,4,7},n=7,i;
        cout<<”\n Original array is :\n”;
        for(i=0;i<n;i++)
                cout<<a[i]<<”, “;

        insert_sort(a,n);
        cout<<"\nThe sorted array is:\n";
        for(i=0; i<n; i++)
                cout<<a[i]<<", ";
}
```

Output is

```
Original array is
8, 5, 9, 3, 16, 4, 7
The sorted array is
3, 4, 5, 7, 8, 9, 16
```

Bubble Sort

Bubble sort compares a[i] with a[i+1] for all i=0..n-2, if a[i] and a[i+1] are not in ascending order then exchange a[i] with a[i+1] immediately. After each iteration, all elements which are not at their proper position move at least one position towards their right place in the array. The process continues until all elements get their proper place in the array (i.e. algorithm terminates if no exchange occurs in the last iteration)

For example, consider the following unsorted array to be sorted using selection sort

Original array	S[0]	**S[1]**	**S[2]**	**S[3]**	**S[4]**	**S[5]**	**S[6]**
	8	**5**	**9**	**3**	**16**	**4**	**7**
After 1st iteration	**5**	**8**	**3**	**9**	**4**	**7**	16
After 2nd iteration	**5**	**3**	**8**	**4**	**7**	9	16
After 3rd iteration	**3**	**5**	**4**	**7**	8	9	16
After 4th iteration	**3**	**4**	**5**	7	8	9	16
After 5th iteration	**3**	**4**	5	7	8	9	16
After 6th iteration	3	4	5	7	8	9	16

```
#include<iostream.h>
void insert_sort(int s[ ],int n)        //n is the no of elements present in the array
{
        int i, j,p;
        for (i=0;  i < n;  i++)
        {
                for( j=1; j < n-1; j++)
                {
                        if(  s[j] > s[j+1)          // consecutive elements are being checked
                        {         // swapping the consecutive elements if first value is > next value
                                p  =  s[j]  ;
                                s[j]= s[j+1] ;
                                s[j+1]  = p ;
                        }
        }
}
void main( )
{
        int a[7]={8,5,9,3,16,4,7},n=7,i;
        cout<<"\n Original array is :\n";
        for(i=0;i<n;i++)
                cout<<a[i]<<", ";

        insert_sort(a,n);

        cout<<"\nThe sorted array is:\n";
        for(i=0; i<n; i++)
                cout<<a[i]<<", ";
}
Output is
Original array is
8, 5, 9, 3, 16, 4, 7
The sorted array is
3, 4, 5, 7, 8, 9, 16
```

Merging of two sorted arrays into third array in sorted order

Algorithm to merge arrays a[m](sorted in ascending order) and b[n](sorted in descending order) into third array C[n+m] in ascending order.

```
#include<iostream.h>
Merge(int a[ ], int m, int b[n], int c[ ])          // m is size of array a and n is the size of array b
{
        int i=0;            // i points to the smallest element of the array a which is at index 0
        int j=m-1;       // j points to the smallest element of the array b which is at the index m-1 since
        b
                        // is sorted in descending order
        int k=0;          //k points to the first element of the array c

        while(i<m && j>=0)
        {
                if(a[i] < b[j])
                        c[k++]=a[i++]; // copy from array a into array c and then increment i and k
                else
                        c[k++]=b[j--]; // copy from array b into array c and then decrement j and
                                        //increment k
        }
```

```
            while(i<m)                    //copy all remaining elements of array a
                    c[k++]=a[i++];
            while(j>=0)                   //copy all remaining elements of array b
                    c[k++]=b[j--];
    }

    void main()
    {
            int a[5]={2,4,5,6,7},m=5;             //a is in ascending order
            int b[6]={15,12,4,3,2,1},n=6;         //b is in descending order
            int c[11];
            merge(a, m, b, n, c);
            cout<<"The merged array is :\n";
            for(int i=0; i<m+n; i++)
                    cout<<c[i]<", ";
    }
```

Output is

```
The merged array is:
1, 2, 2, 3, 4, 4, 5, 6, 7, 12, 15
```

Two dimensional arrays

1	**2**	**3**
4	**5**	**6**

Row-major order (used in C++ to store matrix)

In row-major storage, a multidimensional array in linear memory is accessed such that rows are stored one after the other.

Memory allocation for above matrix will be

1	2	3	4	5	6

1st row 2nd row

offset = row * Numcols + column

Example 1.

For a given array A[10][20] is stored in the memory along the row with each of its elements occupying 4 bytes. Calculate address of A[3][5] if the base address of array A is 5000.

Solution:

For given array A[M][N] where M=Number of rows, N =Number of Columns present in the array

address of A[I][J] = base address + sizeof (type)* offset

address of A[I][J] = base address + sizeof (type)* (I * N + J)

here M=10, N=20, I=3, J=5, sizeof(type) = 4 bytes

address of A[3][5] = 5000 + 4 * (3 * 20 + 5)

= 5000 + 4*65 = 5000+260 = 5260

Example 2.

An array A[50][20] is stored in the memory along the row with each of its elements occupying 8 bytes. Find out the location of A[5][10], if A[4][5] is stored at 4000.

Solution:

Calculate base address of A i.e. address of A[0][0]

For given array A[M][N] where M=Number of rows, N =Number of Columns present in the array

address of A[I][J] = base address + sizeof (type) * (I * N + J)

here M=50, N=20, sizeof(type)=8, I=4, J=5

address of A[4][5] = base address + 8 * (4*20 +5)

4000 = base address + 8 *85

Base address = 4000 – 85 * 8 = 4000 – 680 = 3320

Now to find address of A[5][10]

here M=50, N=20, sizeof(type)=8, I=5, J=10

Address of A[5][10] = base address + 8* (5*20 + 10)

= 3320 + 8*110 = 3320 + 880 = 4200

Column major order (used in Fortran to store matrix)

In column-major storage, a multidimensional array in linear memory is accessed such that columns are stored one after the other.

Memory allocation for above matrix will be

1	4	2	5	3	6

1st column 2nd olumn 3rd column

offset = row + column*NUMROWS

Address of element A[row][column] :-

Example1.

For a given array A[10][20] is stored in the memory along the column with each of its elements occupying 4 bytes. Calculate address of A[3][5] if the base address of array A is 5000.

Solution:

For given array A[M][N] where M=Number of rows, N =Number of Columns present in the array

Address of A[I][J]= base address + sizeof(type) * (J * M + I)

here M=10, N=20, I=3, J=5, sizeof(type)=4 bytes

Address of A[3][5] = 5000 + 4 * (5 * 10 + 3)

= 5000 + 4*53 = 5000 + 215 = 5215

Example2.

An array A[50][20] is stored in the memory along the column with each of its elements occupying 8 bytes. Find out the location of A[5][10], if A[4][5] is stored at 4000.

Solution:

Calculate base address of A i.e. address of A[0][0]

For given array A[M][N] where M=Number of rows, N =Number of Columns present in the array

address of A[I][J]= base address+ sizeof(type) * (J * M + I)

here M=50, N=20, sizeof (type) = 8, I = 4, J = 5

address of A[4][5] = base address +8* (5 * 50 +4)

4000 = base address + 8*254

Base address = 4000-8*55 = 4000 – 2032 = 1968

Now to find address of A[5][10]

here M=50, N=20, sizeof(type)=8, I =5, J=10

Address of A[5][10] = base address + 8 * (10*50 + 10)

=1968 + 8*510 = 1968+4080 = 6048

4 Marks Questions

1. Write a function in C++ which accepts an integer array and its size as arguments and replaces elements having even values with its half and elements having odd values with twice its value

2. Write a function in C++ which accepts an integer array and its size as argument and exchanges the value of first half side elements with the second half side elements of the array.
   Example: If an array of eight elements has initial content as 2,4,1,6,7,9,23,10. The function should rearrange the array as 7,9,23,10,2,4,1,6.

3. Write a function in c++ to find and display the sum of each row and each column of 2 dimensional array. Use the array and its size as parameters with int as the data type of the array.

4. Write a function in C++, which accepts an integer array and its size as parameters and rearrange the array in reverse. Example if an array of five members initially contains the elements as 6,7,8,13,9,19 Then the function should rearrange the array as 19,9,13,8,7,6

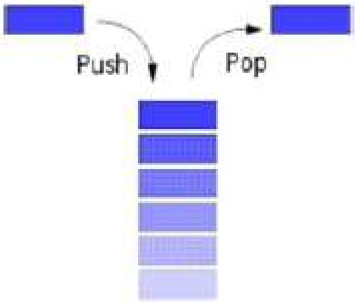
5. Write a function in C++, which accept an integer array and its size as arguments and swap the elements of every even location with its following odd location. Example : if an array of nine elements initially contains the elements as 2,4,1,6,5,7,9,23,10 Then the function should rearrange the array as 4,2,6,1,7,5,23,9,10

6. Write a function in C++ which accepts an integer array and its size as arguments and replaces elements having odd values with thrice and elements having even values with twice its value. Example: If an array of five elements initially contains the elements 3,4,5,16,9 Then the function should rearrange the content of the array as 9,8,15,32,27

STACKS, QUEUES AND LINKED LIST

Stack

In computer science, a stack is a last in, first out (LIFO) *data structure*. A stack can is characterized by only two fundamental operations: *push* and *pop*. The push operation adds an item to the top of the stack. The pop operation removes an item from the top of the stack, and returns this value to the caller.



Stack using array

```
#include<iostream.h>
const int size = 5
class stack
{
          int a[size];                    //array a can store maximum 5 item of type int of the stack
          int top;                        //top will point to the last item pushed onto the stack
public:
          stack( )          //constructor to create an empty stack, top=-1 indicate that no item is
          {top = -1 ;}      //present in the array

          void push(int item)
          {
                    If(top==size-1)
                              cout<<"stack is full, given item cannot be added";
                    else
                              a[++top]=item;          //increment top by 1 then item at new position of the top
                                                      in //the array a
          }
          int pop()
          {
                    If (top==-1)
                    {
                              out<<"Stack is empty ";
                             return -1; //-1 indicates empty stack
                    }
                    else
                         return a[top--];//return the item present at the top of the stack then decrement top
          by 1
          }
};
void main()
{
          stack s1;
          s1.push(3);
          s1.push(5);
```

```
            cout<<s1.pop()<<endl;
            cout<<s1.pop()<<endl;
            cout<<s1.pop();
    }
    Output is
    5
    3
    Stack is empty -1
```

Linked list

In Computer Science, a **linked list** (or more clearly, "singly-linked list") is a data structure that consists of a sequence of nodes each of which contains data and a pointer which points (i.e., a *link*) to the next node in the sequence.

Stack implementation using linked list

```
#include<iostream.h>
struct node
{
        int item;               //data that will be stored in each node
        node * next;            //pointer which contains address of another node
};      //node is a self-referential structure which contains reference of another object type node
class stack
{
        node *top;
public:
        stack()         //constructor to create an empty stack by initializing top with NULL
        { top=NULL; }
        void push(int item);
        int pop();
        ~stack();
};
void stack::push(int item)              //to insert a new node at the top of the stack
{
        node *t=new node;               //dynamic memory allocation for a new object of node type
        if(t==NULL)
        cout<<"Memory not available, stack is full";
        else
        {
        t->item = item;
        t->next = top;          //newly created node will point to the last inserted node or NULL if
        //stack is empty
        top=t;                  //top will point to the newly created node
        }
}

int stack::pop()        //to delete the last inserted node(which is currently pointed by the top)
```

```
{
        if(top==NULL)
        {
        cout<<”Stack is empty \n”;
        return 0;               // 0 indicating that stack is empty
        }
        else
        {
        node *t=top;            //save the address of top in t
        int r=top->item;        //store item of the node currently pointed by top
        top=top->next;          // move top from last node to the second last node
        delete t;               //remove last node of the stack from memory
        return r;
        }
}

stack::~stack()                 //de-allocated all undeleted nodes of the stack when stack goes out of
scope
{
        node *t;
        while(top!=NULL)
        {
        t=top;
        top=top->next;
        delete t;
        }
};

void main()
{
        stack s1;
        s1.push(3);
        s1.push(5);
        s1.push(7);
        cout<<s1.pop()<<endl;
        cout<<s1.pop()<<endl;
        cout<<s1.pop()<<endl;
        cout<<s1.pop()<<endl;
}
```

Output is

```
7
5
3
Stack is empty 0
```

Application of stacks in infix expression to postfix expression conversion

Infix expression	**operand1 operator operand2**	for example	**a+b**
Postfix expression	**operand1 operand2 operator**	for example	**ab+**
Prefix expression	**operator operand1 operand2**	for example	**+ab**

Some example of infix expression and their corresponding postfix expression

Infix expression	postfix expression
a*(b-c)/e	abc-*e/
(a+b)*(c-d)/e	ab+cd-*e/
(a+b*c)/(d-e)+f	abc*+de-/f+

Algorithm to convert infix expression to postfix expression using stack:-

Suppose x is an infix expression and find postfix expression Y

1. Push “(“ to the STACK, and add “)” to the end of X.
2. Scan X from left to right and REPEAT Steps 3 to 6 for each element of X UNTIL the STACK is empty.
3. If an operand is encountered, add it to Y.
4. If a left parenthesis is encountered, push it on to STACK.
5. If an operator is encountered then:

   (a) Repeatedly pop from STACK and add to Y each operator which has the same precedence as or higher precedence than operator.

   (b) Add operator to STACK.
6. If a right parenthesis is encountered. Then:

   (a) Repeatedly pop from the STACK and add to Y each operator until a left parenthesis is encountered.

   (b) Remove the left parenthesis.
7. End

For example convert the infix expression (A+B)*(C-D)/E into postfix expression showing stack status after every step.

Symbol scanned from infix	Stack status	Postfix expression
	(	
(	(**(**	
A	(**(**	A
+	((+	A
B	((+	AB

)	(	AB+
*	(*	AB+
(	(*(	AB+
C	(*(	AB+C
-	(*(-	AB+C
D	(*(-	AB+CD
)	(*	AB+CD-
/	(/	AB+CD-*
E	(/	AB+CD-*E
)		AB+CD-*E/
Answer: Postfix expression of **(A+B)*(C-D)/E** is **AB+CD-*E/**		

Evaluation of Postfix expression using Stack

Algorithm to evaluate a postfix expression P.

```
/*Reading of expression takes place from left to right*/

1.  Read the next element     //First element for the first time
2.  If element is an operator then push the element in the Stack
3.  If the element is an operator then
      {
4.  Pop two operands from the stack                  //pop one operator in case of unary operator
5.  Evaluate the expression formed by the two operands and the operator
6.  Push the result of the expression in the stack.
      }
7.   If no-more-elements then
```

```
            Pop the result
        Else
            Go to step 1.
    8.  End.
```

Example1: Evaluate the following postfix expression showing stack status after every step

8, 2, +, 5, 3, -, *, 4 /

token scanned from postfix expression	Stack status after processing the scanned token	Operation performed
8	**8**	Push 8
2	8, **2**	Push 2

+	**10**	Op2=pop() i.e 2
		Op1=pop() i.e 8
		Push(op1+op2) i.e. 8+2

5	10, **5**	Push(5)
3	10, 5, **3**	Push(3)

-	10, **2**	Op2=pop() i.e. 3
		Op1=pop() i.e. 5
		Push(op1-op2) i.e. 5-3

*	**20**	Op2=pop() i.e. 2
		Op1=pop() i.e. 10
		Push(op1-op2) i.e. 10*2

4	20, **4**	Push 4

/	5	Op2=pop() i.e. 4
		Op1=pop() i.e. 20
		Push(op1/op2) i.e. 20/4

NULL	Final result 5	**Pop 5 and return 5**

Example2: Evaluate the following Boolean postfix expression showing stack status after every step

True, False, True, AND, OR, False, NOT, AND

token scanned from postfix expression	Stack status after processing the scanned token	Operation performed
True	**True**	Push True
False	True, **False**	Push False
True	True, False, **True**	Push True

AND	True, **False**	Op2=pop() i.e. True
		Op1=pop() i.e. False
		Push(Op2 AND Op1) i.e. False ANDTrue=False

OR	True	Op2=pop() i.e. False
		Op1=pop() i.e. True
		Push(Op2 OR Op1) i.e. True OR False=True

False	True, **False**	Push False
NOT	True, **True**	Op1=pop() i.e. False Push(NOT False) i.e. NOT False=True

AND	**True**	Op2=pop() i.e. True
		Op1=pop() i.e. True
		Push(Op2 AND Op1) i.e. True AND True=True

NULL	Final result **True**	Pop True and Return True

QUEUE

Queue is a linear data structure which follows First in First out (FIFO) rule in which a new item is added at the **rear end** and **deletion** of item is from the **front end** of the queue. In a FIFO data structure, the first
element added to the queue will be the first one to be removed. Linear Queue implementation using Array

```
#include<iostream.h>
const int size=5;                // size of queue
class queue
{       int front , rear;
        int a[size];
public:
        queue()                  //Constructor to create an empty queue
        {       front=0;
                rear=0;
        }

        void addQ( )             // insertion in array queue
        {       if(rear==size)
                        cout<<”queue is full<<endl;
                else
                        a[rear++]=item;
        }

        int delQ( )              // deletion from the array queue
        {       if(front==rear)
                {
                        cout<<”queue is empty”<<endl;
                        return 0;
                }
                else
                        return a[front++];
        }
};
void main()
{
        queue q1;
        q1.addQ(3);
        q1.addQ(5) ;
        q1.addQ(7) ;
        cout<<q1.delQ()<<endl ;
        cout<<q1.delQ()<<endl ;
        cout<<q1.delQ()<<endl;
        cout<<q1.delQ()<<endl;
}
```

Output is

```
3
5
```

```
7
Queue is empty 0
```

Queue using linked list

```
#include<iostream.h>
struct node
{       int item;
        node *next;
};
class queue
{       node *front, *rear;
public:
        queue( )                        // constructor to create empty queue
        {       front=NULL;
                rear=NULL;
        }
        void addQ(int item);
        int delQ();
};
void queue::addQ(int item)
{       node * t=new node;
        t->item=item;
        t->next=NULL;
        if (rear==NULL)         //if the queue is empty
        {       rear=t;
                front=t;        //rear and front both will point to the first node
        }
        else
        {       rear->next=t;
                rear=t;
        }
}
int queue::delQ()
{       if(front==NULL)
                cout<<"queue is empty"<<return 0;
        else
        {       node *t=front;
                int r=t->item;
                front=front->next; //move front to the next node of the queue
                if(front==NULL)
                        rear==NULL;
                delete t;
                return r;
        }
}
```

```
void main()
{         queue q1;
          q1.addQ(3);
          q1.addQ(5) ;
          q1.addQ(7) ;
          cout<<q1.delQ()<<endl ;
          cout<<q1.delQ()<<endl ;
          cout<<q1.delQ()<<endl;
          cout<<q1.delQ()<<endl;
}
```

2,3 & 4 Marks Practice Questions

1. Convert the following infix expressions to postfix expressions using stack 2

(i) A + (B * C) ^ D – (E / F – G)

(ii) A * B / C * D ^ E * G / H

(iii) ((A*B)-((C_D)*E/F)*G

2. Evaluate the following postfix expression E given below; show the contents of the stack during the evaluation

(i) E= 5,9,+2,/,4,1,1,3,_,*,+ 2

(ii) E= 80,35,20,-,25,5,+,-,*

(iii) E= 30,5,2,^,12,6,/,+,-

(iv) E=15, 3, 2, +, /, 7, + 2, *

3. An array A[40][10] is stored in the memory along the column with each element occupying 4 bytes. Find out the address of the location A[3][6] if the location A[30][10] is stored at the address 9000.
3

4. Define functions in C++ to perform a PUSH and POP operation in a dynamically allocated stack considering the following : 4

```
struct Node
{
          int X,Y;
          Node *Link;
};
class STACK
{
          Node * Top;
public:
          STACK( )
          { TOP=NULL;}
          void PUSH( );
          void POP( );
          ~STACK( );
};
```

5. Write a function in C++ to perform a Add and Delete operation in a dynamically allocated Queue considering
the following: 4

```
struct node
{
        int empno ;
        char name[20] ;
        float sal ;
        Node *Link;
};
```

UNIT-3 DATABASE AND SQL

Data :- Raw facts and figures which are useful to an organization. We cannot take decisions on the basis of data.

Information:- Well processed data is called information. We can take decisions on the basis of information

Field:- Set of characters that represents specific data element.

Record: Collection of fields is called a record. A record can have fields of different data types.

File: Collection of similar types of records is called a file.

Table: Collection of rows and columns that contains useful data/information is called a table. A table generally refers to the passive entity which is kept in secondary storage device.

Relation: Relation (collection of rows and columns) generally refers to an active entity on which we can perform various operations.

Database: Collection of logically related data along with its description is termed as database.

Tuple: A row in a relation is called a tuple.

Attribute: A column in a relation is called an attribute. It is also termed as field or data item.

Degree: Number of attributes in a relation is called degree of a relation.

Cardinality: Number of tuples in a relation is called cardinality of a relation.

Primary Key: Primary key is a key that can uniquely identifies the records/tuples in a relation. This key can never be duplicated and NULL.

Foreign Key: Foreign Key is a key that is defined as a primary key in some other relation. This key is used to enforce referential integrity in RDBMS.

Candidate Key: Set of all attributes which can serve as a primary key in a relation.

Alternate Key: All the candidate keys other than the primary keys of a relation are alternate keys for a relation.

DBA: Data Base Administrator is a person (manager) that is responsible for defining the data base schema, setting security features in database, ensuring proper functioning of the data bases etc.

Relational Algebra

The relation algebra is the collection of operations on relations. Each operation takes one or more relations (tables) and produces another relation as its result. The operations defined in relational algebra are *select, project, Cartesian product, union, set difference, set interception, natural join, division etc.*

1. **Select operation(denoted by σ):-** select operation is used to select rows from a elation <"

   Let us consider the table **item**

ItemNo	Item_Name	Price
I1	Milk	10
I2	Bread	15
I3	Ice Cream	25
I4	Namkeen	20
I5	Cake	10

2. **Project Operation (denoted by π):-** Project operation select columns from a relation.
   Consider above table **Item**
   To display item name & price of all items from Item table we can write
   $\pi_{\text{Item\_Name, Price}}$ (Item)

Result will be

Item_Name	Price
Milk	10
Bread	15
Ice Cream	25
Namkeen	20
Cake	10

3. **The Cartesian product operation (denoted by** x **):-** the Cartesian product of relation A and B is written as A x B. The Cartesian product yield a new relation having degree (Degree of A + Degree of B) and Cardinality (cardinality of A x Cardinality of B)
   Consider the following table **student** and **instructor**

   The Cartesian product **Student** x **Instructor** result in following relation

Adno	Stu_Name	Passed	**Id**	**Inst_name**	**Subject**
1023	Ajay	Y	101	Manoj	CS
1023	Ajay	Y	102	Subhash	ACC
6151	Sunil	N	101	Manoj	CS
6151	Sunil	N	102	Subhash	ACC
7575	Vinay	y	101	Manoj	CS

7575	Vinay	y	102	Subhash	ACC

4. **The Union Operation (denoted by U):-** it produces a relation that contains tuples from both operand relations.
   Consider the following relations **science** and **commerce**

The result of **Science U Commerce** will be as follows

Adno	**Name**	**Class**
2190	Amit	XII
2345	Nihan	XII
5467	ajay	XI
5423	Sanjay	XII
7665	sumit	XI

5. **The Set Difference Operation (Denoted by -):-** allows to find tuples that are in one relation but not in another relation.
   Consider above relation **science** and **commerce**

The result of **Science - Commerce** will be as follows

Adno	**Name**	**Class**
2190	Amit	XII
5467	ajay	XI

6. **The Set Interception Operation (denoted by ∩) :-**Set Interception operation finds tuples that are common to the two operand relations.
   Consider above relation **science** and **commerce**

The result of **Science ∩ Commerce** will be as follows

Adno	**Name**	**Class**
2345	Nihan	XII

Structured Query Language

SQL is a non-procedural language that is used to create, manipulate and process the databases(relations).

Characteristics of SQL

- It is very easy to learn and use.
- Large volume of databases can be handled quite easily.
- It is non-procedural language. It means that we do not need to specify the procedures to accomplish a task but just to give a command to perform the activity.
- SQL can be linked to most of other high level languages that makes it first choice for the database programmers.

Processing Capabilities of SQL

The following are the processing capabilities of SQL

1. **Data Definition Language (DDL)**
   DDL contains commands that are used to create the tables, databases, indexes, views, sequences and synonyms etc.
   e.g: Create table, create view, create index, alter table etc.
2. **Data Manipulation Language (DML)**
   DML contains command that can be used to manipulate the data base objects and to query the databases for information retrieval.
   e.g Select, Insert, Delete, Update etc.
3. **Data Control Language:**
   This language is used for controlling the access to the data. Various commands like GRANT, REVOKE etc are available in DCL.
4. **Transaction Control Language (TCL)**
   TCL include commands to control the transactions in a data base system. The commonly used commands in TCL are COMMIT, ROLLBACK etc.

Data types of SQL

Support the following data types

Data Type	**Syntax**	**Description**	**Example**
NUMBER	Number(n,d)	• Used to store a numeric value in a field/column • Where **n** specifies the number of digits and **d** specifies the number of digits after the decimal point.	**Amt Number(6,2)**
CHAR	Char (size)	Used to store fixed length string of length size	**Name Char(20)**
VARCHAR / VARCHAR2	varchar(size) / varchar2(size)	Used to store variable length string up to length size	**Address Varchar2(30)**
DATE	DATE	Used to store Date	**DOB Date**
LONG	LONG	This data type is used to store variable length strings of upto 2 GB size	**Accno** LONG

RAW/LONG RAW	RAW(bytes)/ LONG RAW(bytes)	Used to store binary data (images/pictures/animation/clips etc.) up to the size **bytes**	**Address Raw(500)**

1&2 mark questions

Q1. Define the terms:

(i) Database Abstraction

(ii) Data inconsistency

(iii)Conceptual level of database implementation/abstraction

(iv)Primary Key

(v) Candidate Key

(vi)Relational Algebra

(vii) Domain

Ans:. Define the terms:

i. Database Abstraction

Ans: Database system provides the users only that much information that is required by them, and hides certain details like, how the data is stored and maintained in database at hardware level. This concept/process is Database abstraction.

ii. Data inconsistency

Ans: When two or more entries about the same data do not agree i.e. when one of them stores the updated information and the other does not, it results in data inconsistency in the database.

iii. Conceptual level of database implementation/abstraction

Ans: It describes what data are actually stored in the database. It also describes the relationships existing among data. At this level the database is described logically in terms of simple data-structures.

iv. Primary Key

Ans : It is a key/attribute or a set of attributes that can uniquely identify tuples within the relation.

v. Candidate Key

Ans : All attributes combinations inside a relation that can serve as primary key are candidate key as they are candidates for being as a primary key or a part of it.

vi. Relational Algebra

Ans : It is the collections of rules and operations on relations(tables). The various operations are selection, projection, Cartesian product, union, set difference and intersection, and joining of relations.

vii. Domain

Ans : it is the pool or collection of data from which the actual values appearing in a given column are drawn.

2 marks Practice questions

1. What is relation? What is the difference between a tuple and an attribute?
2. Define the following terminologies used in Relational Algebra:
   (i) selection (ii) projection (iii) union (iv) Cartesian product
3. What are DDL and DML?
4. Differentiate between primary key and candidate key in a relation?
5. What do you understand by the terms **Cardinality** and **Degree** of a relation in relational database?
6. Differentiate between DDL and DML. Mention the 2 commands for each category.

Database and SQL : 6 marks questions

1. Write SQL Command for (a) to (d) and output of (g)

TABLE : GRADUATE

S.NO	**NAME**	**STIPEND**	**SUBJECT**	**AVERAGE**	**DIV**
1	KARAN	**400**	PHYSICS	**68**	I
2	DIWAKAR	**450**	COMP Sc	**68**	I
3	DIVYA	**300**	CHEMISTRY	**62**	I
4	REKHA	**350**	PHYSICS	**63**	I
5	ARJUN	**500**	MATHS	**70**	I
6	SABINA	**400**	CHEMISTRY	**55**	II
7	JOHN	**250**	PHYSICS	**64**	I
8	ROBERT	**450**	MATHS	**68**	I
9	RUBINA	**500**	COMP Sc	**62**	I
10	VIKAS	**400**	MATHS	**57**	II

a. List the names of those students who have obtained DIV I sorted by NAME.

b. Display a report, listing NAME, STIPEND, SUBJECT and amount of stipend received in a year assuming that the STIPEND is paid every month.

c. To count the number of students who are either PHYSICS or COMPUTER SC graduates.

d. To insert a new row in the GRADUATE table: 11,"KAJOL", 300, "computer sc", 75, 1

e. Give the output of following sql statement based on table GRADUATE:

(i) Select MIN(AVERAGE) from GRADUATE where SUBJECT="PHYSICS";

(ii) Select SUM(STIPEND) from GRADUATE WHERE div=2;

(iii) Select AVG(STIPEND) from GRADUATE where AVERAGE>=65;

(iv) Select COUNT(distinct SUBJECT) from GRADUATE;

Sol :

a. SELECT NAME from GRADUATE where DIV = 'I' order by NAME;

b. SELECT NAME,STIPEND,SUBJECT, STIPEND*12 from GRADUATE;

c. SELECT SUBJECT,COUNT(*) from GRADUATE group by SUBJECT having SUBJECT='PHYISCS' or SUBJECT='COMPUTER SC';

d. INSERT INTO GRADUATE values(11,'KAJOL',300,'COMPUTER SC',75,1);

e. (i) 63

(ii) 800

(iii) 475

(iv) 4

2. Consider the following tables Sender and Recipient. Write SQL commands for the statements (i) to (iv) and give the outputs for SQL queries (v) to (viii).

Sender

SenderID	SenderName	SenderAddress	City
ND01	R Jain	2, ABC Appls	New Delhi
MU02	H Sinha	12 Newtown	Mumbai
MU15	S Jha	27/A, Park Street	Mumbai
ND50	T Prasad	122-K,SDA	New Delhi

Recipients

RecID	SenderID	RecName	RecAddress	recCity
KO05	ND01	R Bajpayee	5, Central Avenue	Kolkata
ND08	MU02	S Mahajan	116, A-Vihar	New Delhi
MU19	ND01	H Singh	2A, Andheri East	Mumbai
MU32	MU15	P K Swamy	B5, C S Terminals	Mumbai
ND48	ND50	S Tripathi	13, BI D Mayur Vihar	New delhi

(i) To display the names of all Senders from Mumbai

Ans.
```
SELECT sendername from Sender
where sendercity=’Mumbai’;
```

(ii) To display the RecIC, Sendername, SenderAddress, RecName, RecAddress for every Recipient.

Ans.
```
Select R.RecIC, S.Sendername, S.SenderAddress, R.RecName, R.RecAddress
from Sender S, Recepient R
where S.SenderID=R.SenderID ;
```

(iii)To display Recipient details in ascending order of RecName

Ans.
```
SELECT * from Recipent ORDER By RecName;
```

(iv)To display number of Recipients from each city

Ans.
```
SELECT COUNT( *) from Recipient
Group By RecCity;
```

(v) SELECT DISTINCT SenderCity from Sender;

Ans.

SenderCity
Mumbai
New Delhi

(vi) SELECT A.SenderName, B.RecName From Sender A, Recipient B
Where A.SenderID = B.SenderID AND B.RecCity ='Mumbai';

Ans.

A.SenderName	**B.RecName**
R Jain	H Singh
S Jha	P K Swamy

(vii) SELECT RecName, RecAddress From Recipient
Where RecCity NOT IN ('Mumbai', 'Kolkata') ;

Ans.

RecName	**RecAddress**
S Mahajan	116, A Vihar
S Tripathi	13, BID, Mayur Vihar

(viii) SELECT RecID, RecName FROM Recipent
Where SenderID='MU02' or SenderID='ND50';

Ans.

RecID	RecName
ND08	S Mahajan
ND48	STripathi

3. Write SQL command for (a) to (f) on the basis of the table SPORTS

Table: SPORTS

Student NO	Class	Name	Game1	Grade	Game2	Grade2
10	**7**	Sammer	Cricket	**B**	Swimming	**A**
11	**8**	Sujit	Tennis	**A**	Skating	**C**
12	**7**	Kamal	Swimming	**B**	Football	**B**
13	**7**	Venna	Tennis	**C**	Tennis	**A**
14	**9**	Archana	Basketball	**A**	Cricket	**A**
15	**10**	Arpit	Cricket	**A**	Atheletics	**C**

a. Display the names of the students who have grade 'C' in either Game1 or Game2 or both.

b. Display the number of students getting grade 'A' in Cricket.

c. Display the names of the students who have same game for both Game1 and Game2.

d. Display the games taken up by the students, whose name starts with 'A'.

e. Add a new column named 'Marks'.

f. Assign a value 200 for Marks for all those who are getting grade 'B' or grade 'A' in both Game1 and Game2.

Ans : a) SELECT Name from SPORTS where grade='C' or Grade2='C';

b) SELECT Count(*) from SPORTS where grade='A';

c) SELECT name from SPORTS where game1 = game2;
d) SELECT game,game2 from SPORTS where name like ‘A%’;
e) ALTER TABLE SPORTS add (marks int(4));
f) UPDATE SPORTS set marks=200 where grade=’A’;

4. Consider the following tables Stationary and Consumer. Write SQL commands for the statement (i) to (iv) and output for SQL queries (v) to (viii):

Table: Stationary

S_ID	**StationaryName**	**Company**	**Price**
DP01	Dot Pen	ABC	10
PL02	Pencil	XYZ	6
ER05	Eraser	XYZ	7
PL01	Pencil	CAM	5
GP02	Gel Pen	ABC	15

Table: Consumer

C_ID	**ConsumerName**	**Address**	**S_ID**
01	Good Learner	Delhi	PL01
06	Write Well	Mumbai	GP02
12	Topper	Delhi	DP01
15	Write & Draw	Delhi	PL02
16	Motivation	Banglore	PL01

(i) To display the details of those consumers whose Address is Delhi.
(ii) To display the details of Stationary whose Price is in the range of 8 to 15. (Both Value included)
(iii)To display the ConsumerName, Address from Table Consumer, and Company and Price from table Stationary, with their corresponding matching S_ID.
(iv)To increase the Price of all stationary by 2.
(v) SELECT DISTINCT Address FROM Consumer;
(vi) SELECT Company, MAX(Price), MIN(Price), COUNT(*) from Stationary GROUP BY Company;
(vii) SELECT Consumer.ConsumerName, Stationary.StationaryName, Stationary.Price FROM Strionary, Consumer WHERE Consumer.S_ID=Stationary.S_ID;

(viii) Select StationaryName, Price*3 From Stationary;

5. Consider the following tables GARMENT and FABRIC. Write SQL commands for the statements (i) to (iv) and give outputs for SQL queries (v) to (viii).

Table : GARMENT

GCODE	DESCRIPTION	PRICE	FCODE	READYDATE
10023	PENCIL SKIRT	1150	F03	19–DEC–08
10001	FORMAL SHIRT	1250	F01	12–JAN–08
10012	INFORMAL SHIRT	1550	F02	06–JUN–08
10024	BABY TOP	750	F03	07–APR–07
10090	TULIP SKIRT	850	F02	31–MAR–07
10019	EVENING GOWN	850	F03	06–JUN–08
10009	INFORMAL PANT	1500	F02	20–OCT–08
10007	FORMAL PANT	1350	F01	09–MAR–08
10020	FROCK	850	F04	09–SEP–07
10089	SLACKS	750	F03	20–OCT–08

Table : FABRIC

FCODE	TYPE
F04	POLYSTER
F02	COTTON
F03	SILK
F01	TERELENE

(i) To display GCODE and DESCRIPTION of a each dress in descending order of GCODE.

(ii) To display the details of all the GARMENTs, which have READYDATE in between 08–DEC–07 and 16–JUN–08 (inclusive of both the dates).

(iii)To display the average PRICE of all the GARMENTs, which are made up of FABRIC with FCODE as F03.

(iv)To display FABRIC wise highest and lowest price of GARMENTs from DRESS table. (Display FCODE of each GARMENT along with highest and lowest price)

(v) SELECT SUM (PRICE) FROM GARMENT WHERE FCODE= ‘F01’;

(vi)SELECT DESCRIPTION, TYPE FROM GARMENT, FABRIC WHERE GARMENT.FCODE = FABRIC. FCODE AND GARMENT. PRICE>=1260;

(vii) SELECT MAX (FCODE) FROM FABRIC;

(viii) SELECT COUNT (DISTINCT PRICE) FROM FABRIC;

UNIT-4 BOOLEAN LOGIC

Low Order Thinking Questions: (Boolean Algebra)

1. State and verify absorption law in Boolean algebra.

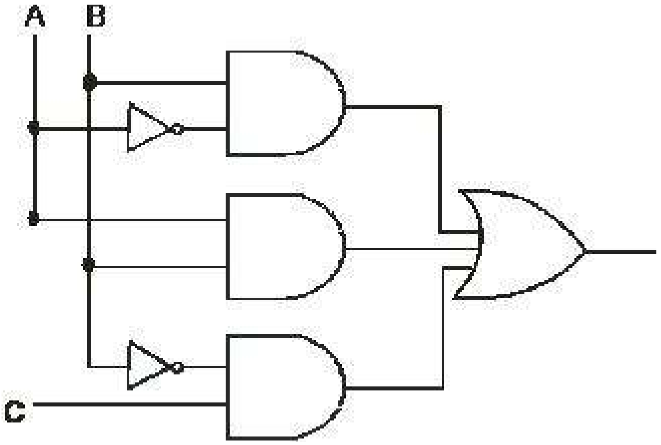
Ans. Absorption Law states that :

a. X+XY=X b. X(X+Y)=X

2. Verify X’.Y+X.Y’=(X’+Y’).(X+Y) algebraically.

Ans. LHS = X’Y + XY’
= (X’+X) (X’+Y’) (Y+X) (Y+Y’)
= 1.(X’+Y’) (X+Y).1
= (X’+Y’) (X+Y)
= RHS, hence proved

3. Write the equivalent Boolean Expression F for the following circuit diagram :



Ans.: A’B+AB+B’C

4. If F(P,Q,R,S) = Π (3,4,5,6,7,13,15) , obtain the simplified form using K-Map.

Ans.:

	R+S	R+S’	R’+S’	R’+S
P+Q			0	
P+Q’	0	0	0	0
P’+Q’		0	0	
P’+Q				

Reduction of groups following the reduction rule :

Quad1 = M4.M5.M6.M7 = P+Q’
Quad2 = M5.M7.M13.M15 = Q’+S’
Pair = M3.M7 = P+R’+S’
Therefore POS of F(P,Q,R,S) = (P+Q’)(Q’+S’)(P+R’+S’)

5. F(a,b,c,d)=Σ(0,2,4,5,7,8,10,12,13,15)
F(a,b,c,d) = B1 + B2 + B3

B1 = m0+m4+m12+m8 = c’d’

B2 = m5+m7+m13+m15 = bd

B3 = m0+m2+m8+m10 = b’d’

F(a,b,c,d) = c’d’ + bd + b’d’

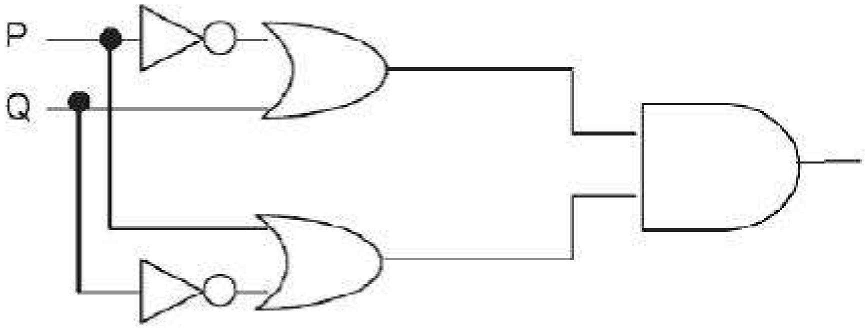
6. Write the equivalent Boolean expression for the following logic circuit:

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

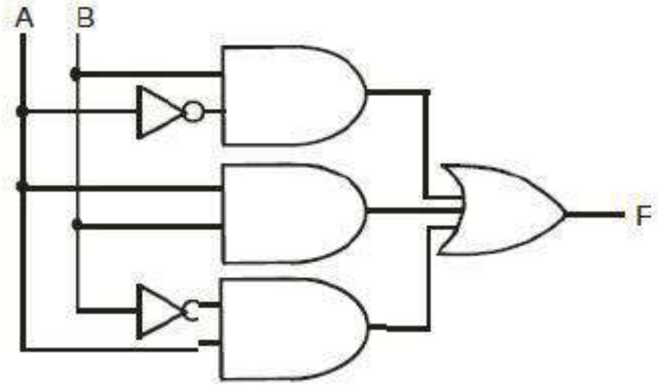
- Express in the product of sums form, the Boolean function F(X,Y,Z), the truth table for which is given below:

1/2 Marks Practice Questions:

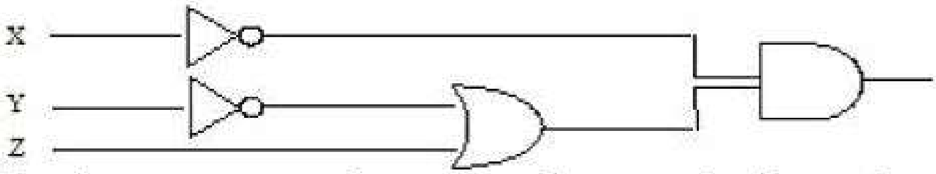
1. State and Prove DeMorgan Law using Truth Table
2. State and prove Absorption Law algebraically.
3. State and Prove Distributive Law algebraically.
4. Write the equivalent Boolean Expression for the following Logic Circuit 2



5. Write the equivalent Boolean Expression F for the following circuit diagram : 2



6. Write the equivalent Boolean Expression F for the following circuit diagram : 2

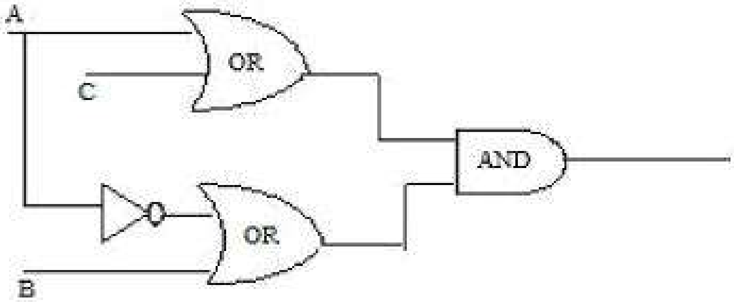


7. Convert the following Boolean expression into its equivalent Canonical Sum of Product Form((SOP)
(X’+Y+Z’).(X’+Y+Z).(X’+Y’+Z).(X’+Y’+Z’) 1

8. Convert the following Boolean expression into its equivalent Canonical Product of Sum form (POS):
A.B’.C + A’.B.C +A’.B.C’ 1

9. Draw a Logical Circuit Diagram for the following Boolean expression:
A.(B+C’) 2

10. Write the equivalent Boolean Expression F for the following circuit diagram: 2



11. Prove that XY+YZ+YZ’=Y algebraically 2
12. Design (A+B).(C+D) using NOR Gate. 2

3 Marks Practice Questions

13. If F(a,b,c,d)=Σ(0,2,4,5,7,8,10,12,13,15), obtain the simplified form using K-Map.
14. If F(a,b,c,d) =Σ(0,3,4,5,7,8,9,11,12,13,15), obtain the simplified form using KMap
15. Obtain a simplified form for a boolean expression
F(U,V,W,Z)= π (0,1,3,5,6,7,10,14,15)
16. Reduce the following boolean expression using K-Map
F(A,B,C,D) = Σ(5,6,7,8,9,12,13,14,15)

UNIT 5 : COMMUNICATION AND NETWORK CONCEPTS

Network

□ The collection of interconnected computers is called a computer network.

□ Two computers are said to be interconnected if they are capable of sharing and exchanging information.

Need

□ Resource Sharing

□ Reliability

□ Cost Factor

□ Communication Medium

Resource Sharing means to make all programs, data and peripherals available to anyone on the network irrespective of the physical location of the resources and the user.

Reliability means to keep the copy of a file on two or more different machines, so if one of them is unavailable (due to some hardware crash or any other) them its other copy can be used.

Cost factor means it greatly reduces the cost since the resources can be shared

Communication Medium means one can send messages and whatever the changes at one end are done can be immediately noticed at another.

Evolution of Networking

1. **ARPANET**:In 1969, The US govt. formed an agency named ARPANET (Advanced Research Projects Agency NETwork) to connect computers at various universities and defense agencies. The main objective of ARPANET was to develop a network that could continue to function efficiently even in the event of a nuclear attack.

2. **Internet (INTERconnection NETwork)**: The Internet is a worldwide network of computer networks. It is not owned by anybody.

3. **Interspace**:InterSpace is a client/server software program that allows multiple users to communicate online with real – time audio, video and text chat in dynamic 3D environments.

SWITCHING TECHNIQUES

Switching techniques are used for transmitting data across networks.

Different types are:

1. **Circuit Switching**: In the Circuit Switching technique, first, the complete end-to-end transmission path between the source and the destination computers is established and then the message is transmitted through the path. The main advantage of this technique is guaranteed delivery of the message. Mostly used for voice communication.

2. **Message Switching**: In the Message switching technique, no physical path is established between sender and receiver in advance. This technique follows the store and forward mechanism.

3. **Packet Switching**: In this switching technique fixed size of packet can be transmitted across the network

Comparison between the Various Switching Techniques: Criteria	**Circuit Switching**	**Message Switching**	**Packet Switching**
Path established in advance	Yes	No	No
Store and forward technique	No	Yes	Yes
Message follows multiple routes	No	Yes	Yes

DATA COMMUNICATION TERMINOLOGIES

Data channel: - The information / data carry from one end to another in the network by channel.

Baud & bits per second (bps) :- It's used to measurement for the information carry of a communication channel. **Measurement Units: - bit**

1 Byte= 8 bits

1 KBPS (Kilo Byte Per Second)= 1024 Bytes
1 Kbps (kilobits Per Second) = 1024 bits
1 Mbps (Mega bits Per Second)=1024 Kbps
Bandwidth: - It is amount of information transmitted or receives per unit time.

Transmission media:

1. Twisted pair cable: - It consists of two identical 1 mm thick copper wires insulated and twisted together. The twisted pair cables are twisted in order to reduce crosstalk and electromagnetic induction.

Advantages:

(i) It is easy to install and maintain.
(ii) It is very inexpensive

Disadvantages:

(i) It is incapable to carry a signal over long distances without the use of repeaters.
(ii) Due to low bandwidth, these are unsuitable for broadband applications.

2. Co-axial Cables: It consists of a solid wire core surrounded by one or more foil or braided wire shields, each separated from the other by some kind of plastic insulator. It is mostly used in the cable wires.

Advantages:

(i) Data transmission rate is better than twisted pair cables.
(ii) It provides a cheap means of transporting multi-channel television signals around metropolitan areas.

Disadvantages:

(i) Expensive than twisted pair cables.
(ii) Difficult to manage and reconfigure.

3. Optical fiber: - An optical fiber consists of thin glass fibers that can carry information in the form of visible light.

Advantages:

(i) **Transmit data over long distance with high security**.
(ii) **Data transmission speed is high**
(iii) Provide better noise immunity
(iv) Bandwidth is up to 10 Gbps.

Disadvantages:

(i) Expensive as compared to other guided media.
(ii) Need special care while installation?

4. Infrared: - The infrared light transmits data through the air and can propagate throughout a room, but will not penetrate walls. It is a secure medium of signal transmission. The infrared transmission has become common in TV remotes, automotive garage doors, wireless speakers etc.

5. Radio Wave: - Radio Wave an electromagnetic wave with a wavelength between 0.5 cm and 30,000m. The transmission making use of radio frequencies is termed as radio-wave transmission

Advantages:

(i) Radio wave transmission offers mobility.
(ii) It is cheaper than laying cables and fibers.
(iii) It offers ease of communication over difficult terrain.

Disadvantages:
(i) Radio wave communication is insecure communication.
(ii) Radio wave propagation is susceptible to weather effects like rains, thunder storms etc.
6. Microwave Wave: - The Microwave transmission is a line of sight transmission. Microwave signals travel at a higher frequency than radio waves and are popularly used for transmitting data over long distances.
Advantages:
(i) It is cheaper than laying cable or fiber.
(ii) It has the ability to communicate over oceans.
Disadvantages:
(i) Microwave communication is an insecure communication.
(ii) Signals from antenna may split up and transmitted in different way to different antenna which leads to reduce to signal strength.
(iii) Microwave propagation is susceptible to weather effects like rains, thunder storms etc.
(iv) Bandwidth allocation is extremely limited in case of microwaves.
7. Satellite link: - The satellite transmission is also a kind of line of sight transmission that is used to transmit signals throughout the world.
Advantages:
(i) Area covered is quite large.
(ii) No line of sight restrictions such as natural mountains, tall building, towers etc.
(iii) Earth station which receives the signals can be fixed position or relatively mobile.
Disadvantages:-
(i) Very expensive as compared to other transmission mediums.
(ii) Installation is extremely complex.
(iii) Signals sent to the stations can be tampered by external interference

Network devices:
Modem: A MODEM (MOdulator DEModulator) is an electronic device that enables a computer to transmit data over telephone lines. There are two types of modems, namely, internal modem and external modem.
RJ45 connector: - The RJ-45(Registered Jack) connectors are the plug-in devices used in the networking and telecommunications applications. They are used primarily for connecting LANs, particularly Ethernet.
Ethernet Card: - It is a hardware device that helps in connection of nodes within a network.
Hub: A hub is a hardware device used to connect several computers together. Hubs can be either active or passive. Hubs usually can support 8, 12 or 24 RJ45 ports.
Switch: A switch (switching hub) is a network device which is used to interconnect computers or devices on a network. It filters and forwards data packets across a network. The main difference between hub and switch is that hub replicates what it receives on one port onto all the other ports while switch keeps a record of the MAC addresses of the devices attached to it.
Gateway: A gateway is a device that connects dissimilar networks.
Repeater: A repeater is a network device that amplifies and restores signals for long distance transmission.

Network topologies and types

Topology :
- Topology refers to the way in which the workstations attached to the network are interconnected.

The BUS Topology: - The bus topology uses a common single cable to connect all the workstations. Each computer performs its task of sending messages without the help of the central server. However, only one workstation can transmit a message at a particular time in the bus topology.
Advantages:

(i) Easy to connect and install.
(ii) Involves a low cost of installation time.
(iii) Can be easily extended.

Disadvantages:-
(i) The entire network shuts down if there is a failure in the central cable.
(ii) Only a single message can travel at a particular time.
(iii) Difficult to troubleshoot an error.

The STAR Topology: - A STAR topology is based on a central node which acts as a hub. A STAR topology is common in homes networks where all the computers connect to the single central computer using it as a hub.

Advantages:
(i) Easy to troubleshoot
(ii) A single node failure does not affect the entire network.
(iii) Fault detection and removal of faulty parts is easier.
(iv) In case a workstation fails, the network is not affected.

Disadvantages:-
(i) Difficult to expand.
(ii) Longer cable is required.
(iii) The cost of the hub and the longer cables makes it expensive over others.
(iv) In case hub fails, the entire network fails.

The TREE Topology: - The tree topology combines the characteristics of the linear bus and the star topologies. It consists of groups of star – configured workstations connected to a bus backbone cable.

Advantages:
(i) Eliminates network congestion.
(ii) The network can be easily extended.
(iii) Faulty nodes can easily be isolated from the rest of the network.

Disadvantages:
(i) Uses large cable length.
(ii) Requires a large amount of hardware components and hence is expensive.
(iii) Installation and reconfiguration is very difficult.

Types of Networks:

LAN (Local Area Network): A Local Area Network (LAN) is a network that is confined to a relatively small area. It is generally limited to a geographic area such as writing lab, school or building. It is generally privately owned networks over a distance not more than 5 Km.

MAN (Metropolitan Area Network): MAN is the networks cover a group of nearby corporate offices or a city and might be either private or public.

WAN (Wide Area Network): These are the networks spread over large distances, say across countries or even continents through cabling or satellite uplinks are called WAN.

PAN (Personal Area Network): A Personal Area Network is computer network organized around an individual person. It generally covers a range of less than 10 meters. Personal Area Networks can be constructed with cables or wirelessly.

Network protocol

- A protocol means the rules that are applicable for a network.
- It defines the standardized format for data packets, techniques for detecting and correcting errors and so on.
- A protocol is a formal description of message formats and the rules that two or more machines must follow to exchange those messages.

- E.g. using library books.

Types of protocols are:
1. HTTP
2. FTP
3. TCP/IP
4. SLIP/PPP

- **Hypertext Transfer Protocol** (**HTTP**) is a communications protocol for the transfer of information on the intranet and the World Wide Web. HTTP is a request/response standard between a client and a server. A client is the end-user; the server is the web site.
- **FTP (File Transfer Protocol)** is the simplest and most secure way to exchange files over the Internet. The objectives of FTP are:
- To promote sharing of files (computer programs and/or data).
- To encourage indirect or implicit use of remote computers.
- To shield a user from variations in file storage systems among different hosts.
- To transfer data reliably, and efficiently.
- **TCP/IP (Transmission Control Protocol / Internet Protocol)**

TCP - is responsible for verifying the correct delivery of data from client to server. Data can be lost in the intermediate network. TCP adds support to detect errors or lost data and to trigger retransmission until the data is correctly and completely received.

IP - is responsible for moving packet of data from node to node. IP forwards each packet based on a four byte destination address (the IP number). The Internet authorities assign ranges of numbers to different organizations. The organizations assign groups of their numbers to departments. IP operates on gateway machines that move data from department to organization to region and then around the world.

Telnet-

It is an older internet utility that lets us log on to remote computer system. It also facilitates for terminal emulation purpose. Terminal emulation means using a pc like a mainframe computer through networking.

Wireless/Mobile Computing

Wireless communication is simply data communication without the use of landlines. Mobile computing means that the computing device is not continuously connected to the base or central network.

1. **GSM(Global System for Mobile communication):** it is leading digital cellular system. In covered areas, cell phone users can buy one phone that will work any where the standard is supported. It uses narrowband TDMA, which allows eight simultaneous calls on the same radio frequency.
2. **CDMA(Code Division Multiple Access)**: it is a digital cellular technology that uses spreadspectrum techniques. CDMA does not assign a specific frequency to each user. Instead ,every channel uses the full available spectrum.

3. **WLL(Wireless in Local Loop) :** WLL is a system that connects subscribers to the public switched telephone network using radio signals as a substitute for other connecting media.
4. **Email(Electronic Mail):** Email is sending and receiving messages by computer.
5. **Chat:** Online textual talk in real time , is called Chatting.
6. **Video Conferencing**: a two way videophone conversation among multiple participants is called video conferencing.
7. **SMS(Short Message Service):** SMS is the transmission of short text messages to and from a mobile pone, fax machine and or IP address.
8. **3G and EDGE:** 3G is a specification for the third generation of mobile communication of mobile communication technology. 3G promises increased bandwidth, up to 384 Kbps when a device is stationary.

EDGE(Enhanced Data rates for Global Evolution) is a radio based high speed mobile data standard.

Network Security Concepts:

Viruses: Viruses are programs which replicate and attach to other programs in order to corrupt the executable codes. Virus enters the computer system through an external source and become destructive.

Worms: Worms are also self- replicating programs that do not create multiple copies of itself on one computer but propagate through the computer network. Worms log on to computer systems using the username and passwords and exploit the system.

Trojan horse: - Though it is a useful program, however, a cracker can use it to intrude the computer system in order to exploit the resources. Such a program can also enter into the computer through an email or free programs downloaded through the Internet.

Spams: Unwanted e-mail (usually of a commercial nature sent out in bulk)

Cookies: Cookies are the text messages sent by a web server to the web browser primarily for identifying the user.

Firewall: A firewall is used to control the traffic between computer networks. It intercepts the packets between the computer networks and allows only authorized packets to pass.

Cyber Law: Cyber law refers to all the legal and regulatory aspects of Internet and the World Wide Web.

Cyber Crimes: Cyber crime involves the usage of the computer system and the computer network for criminal activity.

Hacking: Hacking is an unauthorized access to computer in order to exploit the resources.

Web Services:

WWW: The World Wide Web or W3 or simply the Web is a collection of linked documents or pages, stored on millions of computers and distributed across the Internet.

HTML (Hyper Text Markup Language):- HTML is a computer language that describes the structure and behavior of a web page. This language is used to create web pages.

XML (eXtensible Markup Language):- Extensible Markup Language (XML) is a meta language that helps to describe the markup language.

HTTP (Hyper Text Transfer Protocol):- A protocol to transfer hypertext requests and information between servers and browsers.

Domain Names: A domain name is a unique name that identifies a particular website and represents the name of the server where the web pages reside.

URL:- The Uniform Resource Locator is a means to locate resources such as web pages on the Internet. URL is also a method to address the web pages on the Internet. There are two types of URL, namely, absolute URL and relative URL.

Website: A collection of related web pages stored on a web server is known as a website.

Web browser: A software application that enables to browse, search and collect information from the Web is known as Web browser.

Web Servers: The web pages on the Internet are stored on the computers that are connected to the Internet. These computers are known as web servers.

Web Hosting: - Web Hosting or website hosting is the service to host, store and maintain the websites on the World Wide Web.

Web Scripting: - The process of creating and embedding scripts in a web page is known as Web Scripting.

Types of Scripts:-

(i) **Client Side Scripts**: - Client side scripts supports interaction within a webpage. E.g. VB Script, Java Script, PHP (**P**HP‟S **H**ypertext **P**reprocessor).

(ii) **Server Side Scripts**: - Server side scripting supports execution at server – end. E.g. ASP, JSP, PHP

OPEN SOURCE TERMINOLOGIES

■ **Free Software:** The S/W's is freely accessible and can be freely used changed improved copied and distributed by all and payments are needed to make for free S/W.

■ **Open Source Software:** S/w whose source code is available to the customer and it can be modified and redistributed without any limitation .OSS may come free of cost but nominal charges has to pay nominal charges (Support of S/W and development of S/W).

■ **FLOSS (Free Libre and Open Source Software) :** S/w which is free as well as open source S/W. (Free S/W + Open Source S/W).

■ **GNU (G**NU's **N**ot **U**nix**) :** GNU project emphasize on the freedom and its objective is to create a system compatible to UNIX but not identical with it.

■ **FSF (Free Software Foundation) :** FSF is a non –profit organization created for the purpose of the free s/w movement. Organization funded many s/w developers to write free software.

■ **OSI (Open Source Initiative)** : Open source software organization dedicated to cause of promoting open source software it specified the criteria of OSS and its source code is not freely available.

■ **W3C(World Wide Web Consortium)** : W3C is responsible for producing the software standards for World Wide Web.

■ **Proprietary Software:** Proprietary Software is the s/w that is neither open nor freely available, normally the source code of the Proprietary Software is not available but further distribution and modification is possible by special permission by the supplier.

■ **Freeware:** Freeware are the software freely available , which permit redistribution but not modification (and their source code is not available). Freeware is distributed in *Binary Form* (ready to run) without any licensing fees.

■ **Shareware:** Software for which license fee is payable after some time limit, its source code is not available and modification to the software are not allowed.

■ **Localization:** localization refers to the adaptation of language, content and design to reflect local cultural sensitivities .e.g. Software Localization: where messages that a program presents to the user need to be translated into various languages.

■ **Internationalization:** Opposite of localization.

OPEN SOURCE / FREE SOFTWARE

■ **Linux :** Linux is a famous computer operating system . popular Linux server set of program – LAMP(Linux, Apache, MySQL, PHP)

■ **Mozilla :** Mozilla is a free internet software that includes

- a web browser
- an email client
- an HTML editor
- IRC client

■ **Apache server:** Apache web server is an open source web server available for many platforms such as BSD, Linux, and Microsoft Windows etc.

- Apache Web server is maintained by open community of developers of Apache software foundation.

■ **MYSQL :** MYSQL is one of the most popular open source database system. Features of MYSQl :

- Multithreading
- Multi –User
- SQl Relational Database Server
- Works in many different platform

■ **PostgreSQL :** Postgres SQL is a free software object relational database server . PostgresSQL can be downloaded from www.postgressql.org.

■ **Pango :** Pango project is to provide an open source framework for the layout and rendering of internationalized text into GTK + GNOME environment.Pango using Unicode for all of its encoding ,and will eventually support output in all the worlds major languages.

■ **OpenOffice :** OpenOffice is an office applications suite. It is intended to compatible and directly complete with Microsoft office.

OOo Version 1.1 includes:

- Writer (word processor)
- Calc(spreadsheet)
- Draw(graphics program)etc

■ **Tomcat :** Tomcat functions as a servlet container. Tomcat implements the servlet and the JavaServer Pages .Tomcat comes with the jasper compiler that complies JSPs into servlets.

■ **PHP(Hypertext Preprocessor) :** PHP is a widely used open source programming language for server side application and developing web content.

■ **Python: Python** is an interactive programming language originally as scripting language for Amoeba OS capable of making system calls.

■ Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility (like the electricity grid) over a network (typically the Internet). Clouds can be classified as public, private or hybrid.

Tips to solve Questions based on Networking

1. **Where Server should be placed**: Server should be placed in the building where the number of computers is **maximum**.

2. **Suggest a suitable cable layout of connection**: A suitable cable layout can be suggested in the following two ways:-

(i) On the Basis of Server: First the location of the Server is found out. Server is placed in that building where the number of computers are maximum (According to 80 – 20 rule). After finding the server position, each building distance is compared with the Server building directly or indirectly (taking other building in between). The shortest distance is counted whether it is through directly or indirectly.

(ii) On the Basis of Distance from each building: The distance between the each building is compared to all other buildings either directly or indirectly. The shortest distance is counted whether it is directly or through some other building.

3. **Where the following devices be placed**:

(i) **MODEM**:-

(ii) **HUB / SWITCH:- In all the wings**

(iii) **BRIDGE**:

(iv) **REPEATER**: It is used if the distances higher than 70 m. It regenerates data and voice signals.

(v) **ROUTER**: When one LAN will be connected to the other LAN.

1 and 2 Marks Questions

1. What do you mean by a computer network?

Ans:- Computer network is an interconnection of autonomous computers connected together using transmission media.

2. What is the need for networking the computers?

Ans:- 1. Sharing of Information, 2. Reliability, 3. Reduces cost 4. Time saving

3. What is the full form of ARPANET?

Ans:- Advanced Research Projects Agency Network

4. What are various data transmission modes?

Ans:- There are three modes of data transmission

- Simplex
- Half-duplex
- Full-duplex

5. What is the difference between Simplex and half duplex transmission?

Ans:- In simples transmission mode, the data can be transferred in only one direction where as in half duplex transmission mode, the data can be transmitted in both directions but one at a time.

6. What do you mean by MODEM?

Ans:- MODEM stands for MODulatorDEModuator. It is a device that can convert an analog signal into digital signal and vice versa.

7. Define the terms Bandwidth.

Ans:- Bandwidth is the range of frequencies that is available for the transmission of data. Wider the bandwidth of a communication channel, the more data it can transmit in a given period of time.

8. What are various types of transmission media?

Ans:- There are two broad categories of transmission media

- Guided media
- Unguided Media

9. Explain in brief the advantages and disadvantages of Twisted pair Cable.

Ans:- Advantages

- Inexpensive
- Often available in existing phone system
- Well tested and east to get

Disadvantages

- Susceptible to noise (sound, energy etc.)
- Not as durable as coaxial cable
- Does not support high speed

10. What do you mean by communication protocol?

Ans:- A protocol is a set of rules to enable computers to connect with one another and to exchange information with minimum possible error.

11. List various functions of Communication protocol.

Ans:- Data sequencing, Data Formatting, Flow control, Error Control,Connection Establishment and termination,Data Security

12. List commonly used protocols.

Ans:- HTTP, TCT/IP, FTP, SLIP, PPP, SMTP, POP, ICMP

13. What are the main functions of TCP

Ans:- The TCP does the following activities

- It breaks the data into packets that the network
- Verifies that all the packets arrived at the destination
- Reassembles the data

14. What do you mean by network topology?

Ans:- Topology is how the nodes/computers are interconnected together.

15. List various types of Networks.

Ans:- LAN, MAN, WAN

16. Give names of various networking topologies in LAN.

Ans:- 1.Star Topology, 2.Ring topology, 3.Bus topology 4.Mesh Topology

17. Write two advantages and two disadvantages of STAR topology.

Ans:- Advantages of STAR topology

- It is easy to modify and add new computers to a star network without disturbing the rest of the network.
- Troubleshooting a star topology network is easy

Disadvantages

- All the nodes are dependent on the central system. Hub. Failure of hub result in shutting down of whole of the system
- Long cable length is required

18. What is NFS?

Ans:- NFS stands for Network File System. NFS is a protocol that allows a set of computers to access each others files.

19. What are the characteristics of cloud computing?

Ans. Cloud computing exhibits the following key characteristics:

(1) Agility- improves with users' ability to re-provision technological infrastructure resources.

(2) Application programming interface (API) accessibility to software that enables machines to interact with cloud software in the same way that a traditional user interface (e.g., a computer desktop) facilitates interaction between humans and computers. Cloud computing systems typically use Representational State Transfer (REST)-based APIs.

(3) Cost: cloud providers claim that computing costs reduce. A public-cloud delivery model converts capital expenditure to operational expenditure. This purportedly lowers barriers to entry, as infrastructure is typically provided by a third party and does not need to be purchased for one-time or infrequent intensive computing tasks.

(4) Device and location independence enable users to access systems using a web browser regardless of their location or what device they use (e.g., PC, mobile phone). As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect from anywhere.

(5) Maintenance of cloud computing applications is easier, because they do not need to be installed on each user's computer and can be accessed from different places.

4 Marks Questions : Communication and Network Concepts

1. Knowledge Supplement Organization has set up its new centre at Manglore for its office and web based activities. It has four buildings as shown in the diagram below:

Alpha
Gamma
Lambda
Beta

Center to center distance between various buildings

Alpha to Beta	50m
Beta to Gamma	150m
Gamma to Lambda	25m
Alpha to Lambda	170m
Beta to Lambda	125m
Alpha to Gamma	90m

Number of Computers

Alpha	25
Beta	50
Gamma	125
Lambda	10

i) Suggest a cable layout of connections between the buildings

ii) Suggest the most suitable place(i.e building) to house the server of this organization with a suitable reason.

iii) Suggest the placement of the following devices with justification:

  i. Repeater

  ii. Hub/Switch

iv) The organization is planning to link its front office situated in the city in a hilly region where cable connection is not feasible, suggest an economic way to connect it with reasonably high speed?

2. **BHARATH ELECTRONICS COMPANY** in Coimbatore is setting up the network between its different departments located in different blocks. There are 4 blocks named as Meera (M), Tagore (T), Kalidas (K) and Bharathi (B).

Distances between various blocks are given below:

Block B to Block K	100 m
Block B to Block M	200 m
Block B to Block T	400 m
Block K to Block M	300 m

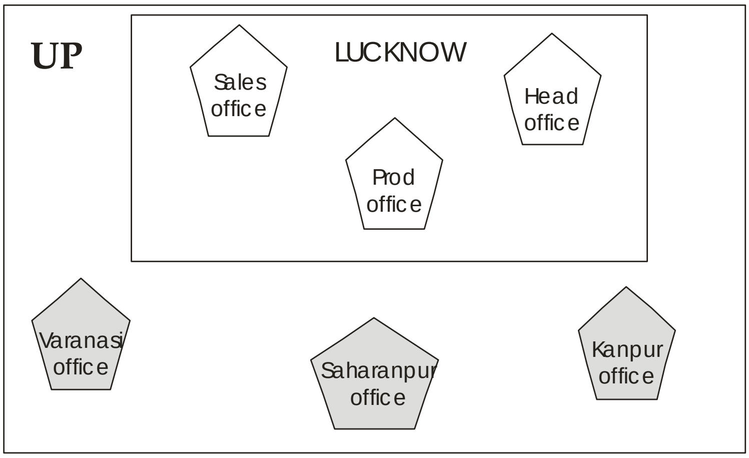
Block M to Block P	100m
Block R to Block P	450 m

a. **Number of Computers:**

Block M	15
Block R	100
Block A	50
Block P	150

i. Suggest a suitable Topology for networking the computers of all Blocks.
ii. Name the Block where the Server is to be installed. Justify your answer.
iii. Suggest the placement of Hub/Switch in the network.
iv. Mention an economic technology to provide Internet accessibility to allBlocks**.**

3. if "Kanganalay Cosmetics" is planning to start their offices in four major cities in Uttar Pradesh to provide cosmetic product support in its retail fields. The company has planned to set up their offices in Lucknow at three different locations and have named them as "Head office", "Sales office", & "Prod office". The company's regional offices are located at Varanasi, Kanpur & Saharanpur. A rough layout of the same is as follows :



An approximate distance between these offices as per network survey team is as follows:

Place from	**Place to**	**Distance**
Head office	Sales office	15 KM
Head office	Prod office	8 KM
Head office	Varanasi Office	295 KM
Head office	Kanpur Office	195 KM
Head office	Saharanpur office	408 KM

Number of computers :

Head office	156
Sales office	25
Prod office	56
Varanasi Office	85
Kanpur Office	107
Saharanpur office	105

i) Suggest the placement of the ***repeater*** with justification. Name the branch where the ***server*** should be installed. Justify your answer.

ii) Suggest the ***device*** to be procured by the company for connecting all the computers within each of its offices out of the following devices:

- Modem
- Telephone
- Switch/Hub

iv) The company is planning to link its head office situated in Lucknow with the office at Saharanpur. Suggest an economic way to connect it; the company is ready to compromise on the speed of connectivity. Justify your answer.

4. Dr. Rizvi Education Society of India is starting its new CBSE School in Mumbai (Maharashtra). The society is already running a School in Jaunpur (UP) named Dr. Rizvi Learners' Academy, having 3 major buildings in 2 km area campus. As a network expert you need to suggest the network plan as per E1 to E4:

Wire Distance Between Various Buildings:

Library building to Admin building	90m
Library building to Academic building	80m
Academic building to Admin building	15m
Jaunpur School to Mumbai School	1350km

Expected number of Computers to be installed in various buildings:

Library Building	20
Academic building	150
Admin building	35
Mumbai School	5

E1. Suggest the cable layout among various buildings inside school campus for connecting the buildings.

E2. Suggest the most suitable place to house the server of the school with a suitable reason.

E3. Suggest an efficient device from the following to be installed in each of the building to connect all the computers:

(i) Bridge (ii) Repeater (iii) Switch

E4. Suggest the most suitable service (very high speed) to provide data connectivity between Rizvi Learners' in Jaunpur and Mumbai CBSE School from the options

HIGHER ORDER THINKING QUESTIONS

Q.1 What is protocol? How many types of protocols are there?

Ans. When computers communicate each other, there needs to be a common set of rules and instructions that each computer follows. A specific set of communication rules is called a protocol. Some protocol: PPP, HTTP, SLIP, FTP, TCP/IP

Q.2 What is the difference between Networking and Remote Networking?

Ans. The main difference between Networking and Remote Networking, is the network which we use in offices or other places locally such LAN or INTERNET and remote networking is one which we use TERMINAL Services to communicate with the remote users such WAN.

Q.3 What is point-to-point protocol?

Ans. A communication protocol used to connect computer to remote networking services include Internet Service Providers. In networking, the Point-to-Point protocol is commonly used to establish a direct connection between two nodes. Its primary use has been to connect computers using a phone line.

Q.4 How gateway is different from router?

Ans. A gateway operates at the upper levels of the OSI model and translates information between two completely different network architectures. Routers allow different networks to communicate with each other. They forward packets from one network to another based on network layer information. A gateway can interpret and translate the different protocols that are used on two distinct networks. Unlike routers that successfully connect networks with protocols that are similar, a gateway perform an application layer conversion of information from one protocol stack to another.

Q.5 What is the role of network administrator?

Ans. Basic tasks for which a network administrator may be responsible:

- Setting up and configuring network hardware and software.
- Installing and configuring network media and connections.
- Connecting user nodes and peripherals of all kinds to the network.
- Adding users to and removing users from the network.
- Managing user account.
- Ensuring the security of the network.
- Provide training to the users to utilize the network's resources.

Q.6 What is the difference between baseband and broadband transmission?

Ans. Baseband is a bi-directional transmission while broadband is a unidirectional transmission.

No Frequency division multiplexing possible in base band but possible in broadband.

SN o	Baseband	Broadband
1	Entire bandwidth of the cable is consumed by a signal	Broadband transmission, signals are sent on multiple frequencies, allowing multiple signals to be sent simultaneously.
2	Digital signals	Analog signals
3	bi-directional transmission	unidirectional transmission

4	No Frequency division multiplexing possible	Frequency division multiplexing possible
5	Uses for short distance	Uses for long distance

Q.7 What are the difference between domain and workgroup?

Ans.

SNo	Domain	Workgroup
1	One or more computers are servers	All Computers are peers.
2	If you have a user account on the domain, you can logon to any computer on the domain.	Each computer has a set of accounts.
3	There can be 100+ computers	Typically not more then 20-30 computers
4	The computers can be on different local network	All computers must be on the same local network.

Q.8 what is the differences between POP3 and IMAP Mail Server?

Ans. IMAP is a standard protocol for accessing e-mail from a local server. A simpler e-mail protocol is Post Office Protocol 3 (POP3), which download mail to the computer and does not maintain the mail on the server. IMAP, e-mails are stored on the server, while in POP3, the messages are transferred to the client's computer when they are read.

Q.9 Name different layer of the ISO OSI Model.

Ans. International Standard Orrganisation – Open Systems Interconnection has seven layers; Physical Layer,Data Link Layer,Network Layer,Transport Layer,Session Layer,Presentation Layer Application Layer

Q.10 What is client server architecture?

Ans. To designated a particular node which is well known and fixed address, to provide a service to the network as a whole. The node providing the service is known as the server and the nodes that use that services are called clients of that server. This type of network is called Client-Server Architecture.

Q.11 What is FDM? Give example.

Ans. FDM-Frequency Division Multiplexing is used in analog transmission. It is often used in short distance. It is code transparent and any terminal of the same speed can use the same sub-channel after the sub-channel is established. The best example if FDM is the way we receive various stations in a radio.

Q.12 Describe the following in brief:

i) MOSAIC ii) USENET iii) WAIS

Ans. i) MOSAIC: is the program for cruising the internet. The National centre wrote this program for Super Computer application at the university of Illinois. It has a simple window interface, which creates useful hypertext links that automatically perform some of the menu bar and button functions.

ii) USENET: is the way to meet people and share information. Usenet newsgroup is a special group set up by people who want to share common interests ranging from current topic to cultural heritages.

iii) WAIS: is a WIDE AREA INFORMATION SERVER.

Computer Science (Code 083)
Sample Paper Set - 1

Max. Marks: 70 **Duration: 3 Hours**

1.

(a) What is the difference between Global Variable and Local Variable? **2**

(b) Write the names of the header files to which the following belong: **1**

(i) strcmp() (ii) fabs()

(c) Rewrite the following program after removing the syntactical errors (if any). Underline each correction. **2**

```
#include [iostream.h]
class PAYITNOW
{
  int Charge;
PUBLIC:
   void Raise(){cin>>Charge;}
   void Show{cout<<Charge;}
};
void main()
{
  PAYITNOW P;
  P.Raise();
  Show();
}
```

(d) Find the output of the following program: **3**

```
#include <iostream.h>
struct PLAY
{ int Score, Bonus;};
void Calculate(PLAY &P, int N=10)
{
     P.Score++;P.Bonus+=N;
}
void main()
{
     PLAY PL={10,15};
     Calculate(PL,5);
     cout<<PL.Score<<”:”<<PL.Bonus<<endl;
     Calculate(PL);
     cout<<PL.Score<<”:”<<PL.Bonus<<endl;
     Calculate(PL,15);
     cout<<PL.Score<<”:”<<PL.Bonus<<endl;
}
```

(e) Find the output of the following program: **2**

```
#include <iostream.h>
#include <ctype.h>
void Encrypt(char T[])
{
      for (int i=0;T[i]!='\0';i+=2)
            if (T[i]=='A' || T[i]=='E') T[i]='#';
            else if (islower(T[i])) T[i]=toupper(T[i]);
                        else T[i]='@';
}
void main()
{
      char Text[]="SaVE EArtH";//The two words in the string Text
                              //are separated by single space
      Encrypt(Text);
      cout<<Text<<endl;
}
```

(f) In the following program, if the value of N given by the user is 15, what maximum and minimum values the program could possibly display? **2**

```
#include <iostream.h>
#include <stdlib.h>
void main()
{
     int N,Guessme;
     randomize();
     cin>>N;
     Guessme=random(N)+10;
     cout<<Guessme<<endl;
}
```

2.

(a) What do you understand by Data Encapsulation and Data Hiding? **2**

a) Answer the questions (i) and (ii) after going through the following class: **2**

```
class Seminar
{
      int Time;
public:
      Seminar()                              //Function 1
      {
      Time=30;cout<<″Seminar starts now″<<end1;
   }
   void Lecture()                            //Function 2
      {
      cout<<″Lectures in the seminar on″<<end1;
   }
      Seminar(int Duration)         //Function 3
      {
      Time=Duration;cout<<″Seminar starts now″<<end1;
   }
   ~Seminar()                                //Function 4
   {
     cout<<″Vote of thanks″<<end1;
   }
};
```

i) In Object Oriented Programming, what is **Function 4** referred as and when does it get invoked/called?

ii) In Object Oriented Programming, which concept is illustrated by **Function 1** and **Function 3** together? Write an example illustrating the calls for these functions.

(c) Define a class TEST in C++ with following description: **4**

Private Members

- **a. TestCode of type integer**
- **b. Description of type string**
- **c. NoCandidate of type integer**
- **d. CenterReqd (number of centers required) of type integer**
- **e. A member function CALCNTR() to calculate and return the number of centers as (NoCandidates/100+1)**

Public Members

- **A function SCHEDULE() to allow user to enter values for TestCode, Description, NoCandidate & call function CALCNTR() to calculate the number of Centres**
- **A function DISPTEST() to allow user to view the content of all the data members**

(d) Answer the questions (i) to (iv) based on the following: **4**

```
class PUBLISHER
{
     char Pub[12];
     double Turnover;
protected:
     void Register();
public:
     PUBLISHER();
     void Enter();
     void Display();
};

class BRANCH
{
     char CITY[20];
protected:
     float Employees;
public:
     BRANCH();
     void Haveit();
     void Giveit();
};

class AUTHOR:private BRANCH,public PUBLISHER
{
     int Acode;
  char Aname[20];
     float Amount;


public:
     AUTHOR();
     void Start();
     void Show();
};
```

(i) Write the names of data members, which are accessible from objects belonging to class AUTHOR.

(ii) **Write the names of all the member functions which are accessible from objects belonging to class BRANCH.**

(iii)**Write the names of all the members which are accessible from member functions of class AUTHOR.**

(iv)**How many bytes will be required by an object belonging to class AUTHOR?**

3.

(a) Write a function in C++ to merge the contents of two sorted arrays A & B into third array C. Assuming array A is sorted in ascending order, B is sorted in descending order, the resultant array is required to be in ascending order.

4

(b) An array S[40][30] is stored in the memory along the row with each of the element occupying 2 bytes, find out the memory location for the element S[20][10], if an element S[15][5] is stored at the memory location 5500. 4

(c) Write a function in C++ to perform Insert operation in a dynamically allocated Queue containing names of students. 4

(d) Write a function in C++ to find the sum of both left and right diagonal elements from a two dimensional array (matrix). 2

(e) Evaluate the following postfix notation of expression: 2
20,30,+,50,40,-,*

4.

(a) Observe the program segment given below carefully and fill the blanks marked as Statement 1 and Statement 2 using seekp() and seekg() functions for performing the required task. 1

```
#include <fstream.h>
class Item
{
   int Ino;char Item[20];
public:
   //Function to search and display the content from a particular
   //record number
   void Search(int );
   //Function to modify the content of a particular record number
   void Modify(int);
};

   void Item::Search(int RecNo)
{
      fstream File;
      File.open("STOCK.DAT",ios::binary|ios::in);
      ______________________
      //Statement 1
      File.read((char*)this,sizeof(Item));
      cout<<Ino<<"==>"<<Item<<endl;
      File.close();
}
void Item::Modify(int RecNo)
{
      fstream File;
      File.open("STOCK.DAT",ios::binary|ios::in|ios::out);
      cout>>Ino;cin.getline(Item,20);
```

```
                  ______________________
//Statement 2
     File.write((char*)this,sizeof(Item));
     File.close();
}
```

(b) Write a function in C++ to count the number of lines present in a text file "STORY.TXT". 2

(c) Write a function in C++ to search for a BookNo from a binary file "BOOK.DAT", assuming the binary file is containing the objects of the following class. 3

```
class BOOK
{
        int Bno;
        char Title[20];
public:
        int RBno(){return Bno;}
        void Enter(){cin>>Bno;gets(Title);}
        void Display(){cout<<Bno<<Title<<endl;}
};
```

5.

(a) What do you understand by Degree and Cardinality of a table? 2

(b) Consider the following tables ACTIVITY and COACH. Write SQL commands for the statements (i) to (iv) and give outputs for SQL queries (v) to (viii) 6

Table: ACTIVITY

ACode	ActivityName	ParticipantsNum	PrizeMoney	ScheduleDate
1001	Relay 100x4	16	10000	23-Jan-2004
1002	High jump	10	12000	12-Dec-2003
1003	Shot Put	12	8000	14-Feb-2004
1005	Long Jump	12	9000	01-Jan-2004
1008	Discuss Throw	10	15000	19-Mar-2004

Table: COACH

PCode	Name	ACode
1	Ahmad Hussain	1001
2	Ravinder	1008
3	Janila	1001
4	Naaz	1003

(i) To display the name of all activities with their Acodes in descending order.

(ii) To display sum of PrizeMoney for each of the Number of participants groupings (as shown in column ParticipantsNum 10,12,16)

(iii) To display the coach's name and ACodes in ascending order of ACode from the table COACH

(iv) To display the content of the GAMES table whose ScheduleDate earliar than 01/01/2004 in ascending order of ParticipantNum.

(v) SELECT COUNT(DISTINCT ParticipantsNum) FROM ACTIVITY;

(vi)SELECT MAX(ScheduleDate),MIN(ScheduleDate) FROM ACTIVITY;

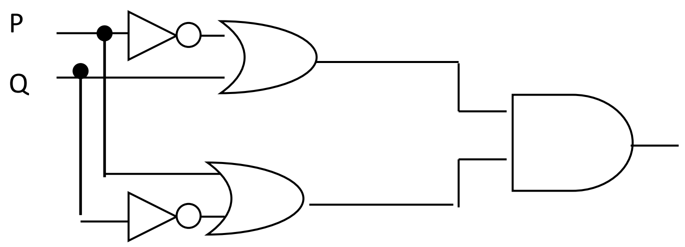
(vii) SELECT SUM(PrizeMoney) FROM ACTIVITY;

(viii) SELECT DISTINCT ParticipantNum FROM COACH;

6.

(a) State and verify Demorgan's Laws. **2**

(b) Write the equivalent Boolean Expression for the following Logic Circuit **2**



(c) Write the POS form of a Boolean function F, which is represented in a truth table as follows: **1**

U	V	W	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

(d) Reduce the following Boolean Expression using K-Map: **3**
F(A,B,C,D)=Σ(0,1,2,4,5,6,8,10)

7.

a) What is the significance of ARPANET in the network? **1**

b) Expand the following terminologies: **1**
(i)CDMA (ii) GSM

c) Give two major reasons to have network security. **1**

d) What is the purpose of using a Web Browser? Name any one commonly used Web Browser. **1**

e) Knowledge Supplement Organisation has set up its new center at Mangalore for its office and web based activities. It has 4 blocks of buildings as shown in the diagram below:

Block A
Block C

Center to center distances between various blocks

Black A to Block B	50 m
Block B to Block C	150 m
Block C to Block D	25 m
Block A to Block D	170 m
Block B to Block D	125 m
Block A to Block C	90 m

Number of Computers

Black A	25
Block B	50
Block C	125
Block D	10

e1) Suggest a cable layout of connections between the blocks. **1**

e2) Suggest the most suitable place (i.e. block) to house the server of this organisation with a suitable reason. **1**

e3) Suggest the placement of the following devices with justification **1**

(i) Repeater

(ii) Hub/Switch

e4) The organization is planning to link its front office situated in the city in a hilly region where cable connection is not feasible, suggest an economic way to connect it with reasonably high speed? **1**

****

CBSE OUTSIDE DELHI 2013-14

(Downloaded from CBSE Academics website)

Series OSR

Code No. 91

Roll No. □□□□□□□

Candidates must write the Code on the title page of the answer-book.

- Please check that this question paper contains **16** printed pages.
- Code number given on the right hand side of the question paper should be written on the title page of the answer-book by the candidate.
- Please check that this question paper contains **7** questions.
- **Please write down the Serial Number of the question before attempting it.**
- 15 minutes time has been allotted to read this question paper. The question paper will be distributed at 10.15 a.m. From 10.15 a.m. to 10.30 a.m., the students will read the question paper only and will not write any answer on the answer-book during this period.

COMPUTER SCIENCE

Time allowed : 3 hours *Maximum Marks : 70*

Instructions :

(i) *All questions are compulsory.*

(ii) *Programming Language : C++*

1. (a) What is the difference between call by reference and call by value with respect to memory allocation ? Give a suitable example to illustrate using C++ code. *2*

(b) Observe the following C++ code and write the name(s) of the header file(s), which will be essentially required to run it in a C++ compiler : *1*

```
void main()
{
  char CH,STR[20];
  cin>>STR;
  CH=toupper(STR[0]);
  cout<<STR<<”starts with”<<CH<<endl;
}
```

(c) Rewrite the following C++ code after removing all the syntax error(s), if present in the code. Make sure that you underline each correction done by you in the code. *2*

Important Note :

- Assume that all the required header files are already included, which are essential to run this code.
- The corrections made by you do not change the logic of the program.

```
typedef char[80] STR;
void main()
{
  Txt STR;
  gets(Txt);
  cout<<Txt[0]<<’\t<<Txt[2];
  cout<<Txt<<endline;
}
```

(d) Obtain the output from the following C++ program as expected to appear on the screen after its execution. *2*

Important Note :

- All the desired header files are already included in the code, which are required to run the code.

```
void main()
{
  char *Text="AJANTA";
  int *P, Num[]={1,5,7,9};
  P=Num;
  cout<<*P<<Text<<endl;
  Text++;
  P++;
  cout<<*P<<Text<<endl;
}
```

(e) Obtain the output of the following C++ program, which will appear on the screen after its execution. *3*

Important Note :

- All the desired header files are already included in the code, which are required to run the code.

```
class Game
{
  int Level, Score;
  char Type;
public:
  Game(char GType='P')
  {Level=1;Score=0;Type=GType;}
  void Play(int GS);
  void Change();
  void Show()
  {
    cout<<Type<<"@"<<Level<<endl;
    cout<<Score<<endl;
  }
};
```

```
void main()
{
  Game A(‘G’),B;
  B.Show();
  A.Play(11);
  A.Change();
  B.Play(25);
  A.Show();
  B.Show();
}
void Game::Change()
{
   Type=(Type==’P’)?’G’:’P’;
}
void Game::Play(int GS)
{
   Score+=GS;
   if(Score>=30)
     Level=3;
   else if(Score>=20)
     Level=2;
   else
     Level=1;
}
```

) Read the following C++ code carefully and find out, which out of the given options (i) to (iv) are the expected correct output(s) of it. Also, write the maximum and minimum value that can be assigned to the variable **Taker** used in the code : *2*

```
void main()
{
  int GuessMe[4]={100,50,200,20};
  int Taker=random(2)+2;
  for (int Chance=0;Chance<Taker;Chance++)
     cout<<GuessMe[Chance]<<"#";
}
```

(i) `100#`

(ii) `50#200#`

(iii) `100#50#200#`

(iv) `100#50`

2. (a) What is function overloading ? Write an example using C++ to illustrate the concept of function overloading. *2*

(b) Answer the questions (i) and (ii) after going through the following class : *2*

```
class Hospital
{
  int Pno, Dno;
public:
  Hospital(int PN);              //Function 1
  Hospital();                    //Function 2
  Hospital(Hospital &H);         //Function 3
  void In();                     //Function 4
  void Disp();                   //Function 5
};
void main()
{
  Hospital H(20);                //Statement 1
}
```

(i) Which of the functions out of Function 1, 2, 3, 4 or 5 will get executed when the Statement 1 is executed in the above code ?

(ii) Write a statement to declare a new object G with reference to already existing object H using Function 3.

(c) Define a class Tourist in C++ with the following specification : *4*

Data Members

- CNo - to store Cab No
- CType - to store a character 'A', 'B', or 'C' as City Type
- PerKM - to store per Kilo Meter charges
- Distance - to store Distance travelled (in KM)

Member Functions

- A constructor function to initialize CType as 'A' and CNo as '0000'
- A function CityCharges( ) to assign PerKM as per the following table :

CType	PerKM
A	20
B	18
C	15

- A function **RegisterCab()** to allow administrator to enter the values for CNo and CType. Also, this function should call **CityCharges()** to assign PerKM Charges.
- A function **Display()** to allow user to enter the value of Distance and display CNo, CType, PerKM, PerKM*Distance (as Amount) on screen.

(d) Consider the following C++ code and answer the questions from (i) to (iv) : *4*

```
class University
{
  long Id;
  char City[20];
protected:
  char Country[20];
public:
  University();
  void Register( );
  void Display( );
};
```

```
class Department: private University
{
   long DCode[10];
  char HOD[20];
protected:
  double Budget;
public:
   Department();
  void Enter();
  void Show();
};
class Student: public Department
{
  long RollNo;
  char Name[20];
public:
   Student();
  void Enroll();
  void View();
};
```

(i) Which type of Inheritance is shown in the above example ?

(ii) Write the names of those member functions, which are directly accessed from the objects of class Student.

(iii) Write the names of those data members, which can be directly accessible from the member functions of class Student.

(iv) Is it possible to directly call function Display( ) of class University from an object of class Department ?

(Answer as Yes or No).

3. (a) Write code for a function `void EvenOdd(int T[], int C)` in C++, to add 1 in all the odd values and 2 in all the even values of the array T. 3

Example : If the original content of the array T is

T[0]	T[1]	T[2]	T[3]	T[4]
35	12	16	69	26

The modified content will be :

T[0]	T[1]	T[2]	T[3]	T[4]
36	14	18	70	28

(b) An array A[20][30] is stored along the row in the memory with each element requiring 4 bytes of storage. If the base address of array A is 32000, find out the location of A[15][10]. Also, find the total number of elements present in this array. 3

(c) Write a user-defined function `AddEnd2(int A[][4],int N,int M)` in C++ to find and display the sum of all the values, which are ending with 2 (i.e., units place is 2). 2

For example if the content of array is :

22	16	12
19	5	2

The output should be

```
36
```

(d) Evaluate the following postfix expression. Show the status of stack after execution of each operation separately : 2

`T, F, NOT, AND, T, OR, F, AND`

(e) Write a function PUSHBOOK( ) in C++ to perform insert operation on a Dynamic Stack, which contains Book_no and Book_Title. Consider the following definition of NODE, while writing your C++ code. 4

```
struct NODE
{
  int Book_No;
  char Book_Title[20];
  NODE *Next;
};
```

4. (a) Fill in the blanks marked as Statement 1 and Statement 2, in the program segment given below with appropriate functions for the required task. *1*

```
class Agency
{
  int ANo;                                //Agent Code
  char AName[20];                         //Agent Name
  char Mobile[12];                        //Agent Mobile
public:
  void Enter();    //Function to enter details of agent
  void Disp();   //Function to display details of agent
  int RAno(){return ANo;}
  void UpdateMobile() //Function to update Mobile
  {
    cout<<"Updated Mobile:";
    gets(Mobile);
  }
};
void AgentUpdate()
{
  fstream F;
  F.open(“AGENT.DAT",ios::binary|ios::in|ios::out);
  int Updt=0;
  int UAno;
  cout<<"Ano (Agent No - to update Mobile):";
  cin>>UAno;
  Agency A;
  while (!Updt && F.read((char*)&A,sizeof(A)))
  {
     if (A.RAno()==UAno)
     {
//Statement 1:To call the function to Update Mobile No.
         ________________________________ ;
//Statement 2:To reposition file pointer to re-write
              the updated object back in the file
        ________________________________ ;
         F.write((char*)&A,sizeof(A));
         Updt++;
      }
  }

  if (Updt)
     cout<<"Mobile Updated for Agent"<<UAno<<endl;
  else
     cout<<"Agent not in the Agency"<<endl;
  F.close();
}
```

(b) Write a function AECount() in C++, which should read each character of a text file **NOTES.TXT**, should count and display the occurrence of alphabets **A** and **E** (including small cases **a** and **e** too). *2*

Example :

If the file content is as follows :

```
CBSE enhanced its
CCE guidelines further.
```

The AECount() function should display the output as

```
A:1
E:7
```

(c) Assuming the class TOYS as declared below, write a function in C++ to read the objects of TOYS from binary file **TOYS.DAT** and display those details of those TOYS, which are meant for children of AgeRange "5 to 8". *3*

```
class TOYS
{
  int ToyCode;
  char ToyName[10];
  char AgeRange;
public:
  void Enter()
  {
    cin>>ToyCode;
    gets(ToyName);
    gets(AgeRange);
  }
  void Display()
  {
    cout<<ToyCode<<":"<<ToyName<<endl;
    cout<<AgeRange<<endl;
  }
  char* WhatAge(){return AgeRange;}
};
```

5. (a) Explain the concept of Cartesian Product between two tables, with the help of appropriate example. *2*

NOTE : Answer the questions (b) and (c) on the basis of the following tables **SHOPPE** and **ACCESSORIES**.

Table : SHOPPE

Id	SName	Area
S001	ABC Computronics	CP
S002	All Infotech Media	GK II
S003	Tech Shoppe	CP
S004	Geeks Tecno Soft	Nehru Place
S005	Hitech Tech Store	Nehru Place

Table : ACCESSORIES

No	Name	Price	Id
A01	Mother Board	12000	S01
A02	Hard Disk	5000	S01
A03	Keyboard	500	S02
A04	Mouse	300	S01
A05	Mother Board	13000	S02
A06	Keyboard	400	S03
A07	LCD	6000	S04
T08	LCD	5500	S05
T09	Mouse	350	S05
T10	Hard Disk	4500	S03

(b) Write the SQL queries : 4

(i) To display Name and Price of all the Accessories in ascending order of their Price.

(ii) To display Id and SName of all Shoppe located in Nehru Place.

(iii) To display Minimum and Maximum Price of each Name of Accessories.

(iv) To display Name, Price of all Accessories and their respective SName where they are available.

(c) Write the output of the following SQL commands : 2

(i) SELECT DISTINCT NAME FROM ACCESSORIES WHERE PRICE >= 5000;

(ii) SELECT AREA, COUNT(*) FROM SHOPPE GROUP BY AREA;

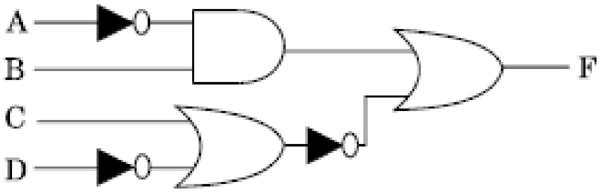
(iii) SELECT COUNT(DISTINCT AREA) FROM SHOPPE;

(iv) SELECT NAME, PRICE*0.05 DISCOUNT FROM ACCESSORIES WHERE SNO IN ('S02', 'S03');

6. (a) Name the law shown below and verify it using a truth table. 2

X+X′ . Y=X+Y

(b) Obtain the Boolean Expression for the logic circuit shown below : 2



(c) Write the Product of Sum form of the function F(X, Y, Z) for the following truth table representation of F : *1*

X	Y	Z	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

(d) Obtain the minimal form for the following Boolean expression using Karnaugh's Map : *3*

$F(A,B,C,D) = \Sigma(1,3,4,5,6,7,12,13)$

7. (a) Write two characteristics of Wi-Fi. *1*

(b) What is the difference between E-mail and Chat ? *1*

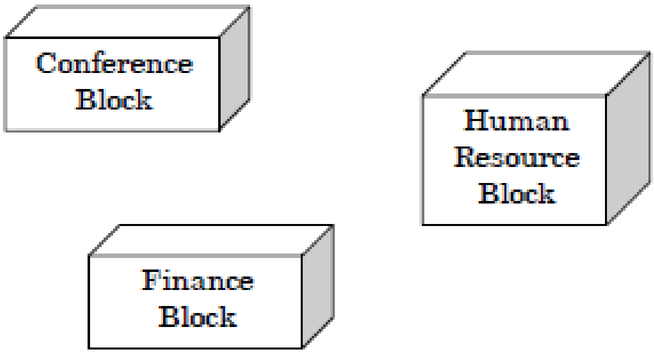
(c) Expand the following : *1*

- GSM
- GPRS

(d) Which type of network (out of LAN, PAN and MAN) is formed, when you connect two mobiles using Bluetooth to transfer a video ? *1*

(e) Tech Up Corporation (TUC) is a professional consultancy company. The company is planning to set up their new offices in India with its hub at Hyderabad. As a network adviser, you have to understand their requirement and suggest to them the best available solutions. Their queries are mentioned as (i) to (iv) below.

Physical Locations of the blocks of TUC



Block to Block distances (in Mtrs.)

Block (From)	Block (To)	Distance
Human Resource	Conference	60
Human Resource	Finance	120
Conference	Finance	80

Expected Number of Computers to be installed in each block

Block	Computers
Human Resource	125
Finance	25
Conference	60

(i) What will the most appropriate block, where TUC should plan to install their server ? *1*

(ii) Draw a block to block cable layout to connect all the buildings in the most appropriate manner for efficient communication. *1*

(iii) What will be the best possible connectivity out of the following, you will suggest to connect the new setup of offices in Bangalore with its London based office ? *1*

- Infrared
- Satellite Link
- Ethernet Cable

(iv) Which of the following devices will be suggested by you to connect each computer in each of the buildings ? *1*

- Gateway
- Switch
- Modem

(f) Write names of any two popular Open Source Software, which are used as Operating Systems. *1*

(g) Write any two important characteristics of Cloud Computing. *1*