Object Oriented Programming Concept Using C++
2 Marks Questions & Answers

Basics of OOPs

1. List out the characteristics of FOP.
1. Large programs are divided into smaller programs known as functions.
2. Most of the functions share global data
3. Functions transform data from one form to another.
4. It employs top-down approach.

2. List out the characteristics of OOP.
1. Programs are divided into objects.
2. Data is hidden.
3. Objects communicate with each other, by sending messages and receiving responses.
4. It follows bottom-up approach.

3. List down the basic concepts of OOP.
1. Objects
2. Classes
3. Data abstraction and encapsulation
4. Inheritance
5. Polymorphism
6. Dynamic binding
7. Message passing

4. Define an object.
Objects are the basic run time entities in an object oriented system.
They may represent a person, a place or any item that a program has to handle.

5. Define Object Oriented Programming.
OOP is a method of implementation in which programs are organized as co-operative collection of objects, each of which represents an instance of some class and whose classes are all members of a hierarchy of classes united through the property called Inheritance.

6. Define a class.
A class is a collection of objects with similar attributes and operations.
Eg. Class – Account
It will create an object savings_account belonging to the class Account.

7. Define Encapsulation.
The wrapping up of data and functions into a single unit is known as encapsulation. The data is kept safe from external interference and misuse.

8. Define Data hiding?
The isolation of data from direct access by the program is called as data hiding or information hiding.

9. Define Abstraction.
It refers to the act of representing essential features without including the background details.

10. Define ADT?
The classes which are using the concept of data abstraction is known as abstract data types (ADT).

11. Define data member and member function?
The attributes which are holding the information is known as data members.
The functions that operate on data member are sometimes called as member function.

12. Define Inheritance?
Inheritance is the process by which objects of one class acquire the properties of objects of another class. The new derived class inherits the members of the base class and also adds its own.

13. Define Polymorphism?
It allows a single name/operator to be associated with different operations depending on the type of data passed to it. An operation may exhibit different behaviors in different instances.

14. Define dynamic binding?
Dynamic binding means that the code associated with the given procedure call is not known until the time of call at runtime.

15. What is message passing?
It is the process of invoking an operation on an object. In response to a message, the corresponding function is executed in the object.

16. List down the benefits of OOP?
1. The principle of data hiding helps to build secure programs.
2. Through inheritance, redundant code is eliminated.
3. Software complexity can be easily managed.

17. List down the applications of OOP.
1. Real time system
2. Simulation and modeling
3. AI and expert system
4. Neural network programming.
5. CAD/CAM systems.

18. What is Implicit Type Conversion?
When an expression consists of data items of different types, the compiler performs type conversions automatically. This is referred as Implicit Type Conversion.

19. What are the differences between 'break' and 'continue' statement.
Break
Break statement takes the control to the outside of loop
It is used in loop and also in switch statement

continue
Continue statement takes the control to the beginning of loop
It can be used only in loop statement.

20. Distinguish 'while' and 'do – while' statements.
While
This is the top tested loop
Loop is not executed if the condition is false
do while
This is the bottom tested loop
Loop is executed at least once even though the condition is false.

21. Give the syntax of Array Initialization with an example.
Data-type array-name[size] = {list of values separated by comma};
(eg.) int mark[5] = {96,45,66,74,82};

22. Give four examples for String Manipulation functions.
strlen( ) – finds the length of string
strcpy( ) – copies the contents of one string to another
strcat( ) – concatenates two strings into one single string
strupr( ) – converts a lower case string to upper case

23. List the different types of parameter passing techniques.
(i) Pass by value
(ii) Pass by Address
(iii) Pass by Reference

24. What is Dynamic memory allocation?
Allocation of memory space for any data structure during the course of the program execution is called as Dynamic memory allocation.

25. How memory management is performed dynamically in C++?
Two operators are available for dynamic memory management.
(i) new – for dynamic memory allocation
(ii) delete – for dynamic memory deallocation

Basics of C++ Language

1. Define class?
A class is a way to bind data and its associated functions together. It allows the data to be hidden if necessary.

2. What are the parts of class specification?
The class specification has two parts
1. Class declaration – To describe the type and scope of its members

2. Class function definition – To describe how the class functions are implemented.

3. Write the syntax of class declaration

class classname
{
Private:
Variable declaration;
Function declaration;
Public:
Variable declaration;
Function declaration;
};

4. Where will you define a member function?

Member functions can be defined in two places.
1. Outside the class definition.
2. Inside the class definition.

5. Give the syntax for member function definition outside the class.

Return type class name:: function name (argument name declaration)
{ function body
}

6. List the characteristics of member function.

1. Member function can access the private data of class.
2. A member function can call another member function directly without using the dot operator.

7. Define nesting of member function.

A member function can be called by using its name inside another member function of the same class. This is known as nesting of member functions.

8. List the properties of static members.

A data member of a class can be qualified as static properties.
1. It is always initialized to zero when the first object of its class is created.
2. It is visible only within the class.

9. Write the properties of static member function.

1. A static function can have access to only other static members declared in the same class.
2. A static member function can be called using the class name (Instead of objects) as follows: class name:: function name;

10. Define constructor?

A constructor is a special member function whose task is to initialize the object of its class. It is called constructor because it constructs the value of data members of the class.

11. What are the characteristics of constructor?
1. Constructors should be declared in public section.
2. They are involved automatically when the objects are created.
3. They do not have return types.
4. They can not be inherited.

12. Define parameterized constructor.
Arguments can be passed to the constructor function when the objects are created. The constructors that can take arguments are called as parameterized constructor.

13. What is an implicit constructor?
C++ compiler has an implicit constructor which creates objects even though it was not defined in the class.

14. What is the use of copy constructor?
Copy constructor is used to declare and initialize an object from another object.
E.g. Class name object2 (object1);
Will define the object2 and at the same time initialize it the values of object1.
15. What do you mean by dynamic construction?
Allocation of memory to objects at the time of their contraction Is known as dynamic contraction of objects.

16. What is the use of destructor?
It is used to destroy the objects that have been created by a constructor. It releases the memory space for future use.

17. What are the characteristics of destructor?
1. A destructor is a member function whose name is the same as the class name but it is preceded by a tilde.
2. It neither takes any argument nor returns any value.
3. It will be invoked implicitly by the compiler to cleanup the storage.

18. Define operator overloading?
The process of making an operator to exhibit different behaviors in different instances is known as operator overloading.

19. Define function overloading?
Performing different types of task using single function name is referred as function overloading.

20. Define Virtual function.
When the form of a member function is changed at run time, that member function is referred to as virtual function.

Unit-I

1.What are the characteristics of procedure oriented programming language?

1. Large programs are divided into smaller programs known as functions
2. Most of the functions share global data
3. Data move openly around the system from function to function
4. Functions transform data from one form to another
5. Uses top-down approach in program design.
6. Concentration is on doing things(algorithms)

2.What are the features of object-oriented programming languages?

1. Programs are divided into objects
2. Data structures designed such that they characterize the objects.
3. Functions that operate on the data of an object are tied together in the data structure
4. Data is hidden and cannot be accessed by external functions
5. Objects may communicate with each other through functions
6. New data and functions can be easily added whenever necessary
7. IT follows bottom-up approach in program design
8. Concentration is on data rather than procedure

3.Define Object Oriented Programming

It is an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand.

4. **What is meant by an object?**

- Objects are the basic run-time entities in an object-oriented system.
- They may represent a person, a place, , a table of data or any item that the program must handle.
- They may also represent user defined data such as vectors, time and lists.
- When a program is executed, the objects interact by sending message to one another.
- Each object contains data and code to manipulate the data.
- Objects can interact without knowing having to know details of each other's data or code.

5. What is meant by Classes?

1. A class is a collection of objects of same type.
2. Once the class has been defined , we can create any number of objects belonging to that class.
3. Each object is associated with the data of type class with which they are created.

6.What is meant by Encapsulation?

- The wrapping up of data and functions into a single unit (called class) is known as encapsulation
- Data encapsulation is the most striking feature of a class.
- The data is not accessible to the outside world and only those functions which are wrapped in the class can access it.
- These functions provide the interface between the object's data and the program
- This insulation of the data from direct access by the program is called data hiding.

7. **What is meant by Abstraction**

- => It refers to the act of representing essential features without including the background details or explanation
- Classes use the concept of abstraction and are defined as a list of abstraction and are defined as a list of abstract attributes and functions to operate on these attributes.
- They encapsulate all the essential properties of the objects that are to be created

8. Define Inheritance.

- Inheritance is the process by which objects of one class acquire the properties of objects of another class.
- It provides the idea of reusability.
- This is possible by deriving a new class from the existing class . The new class will have the combined features of both the classes.

9.Define polymorphism

- It means the ability to take more than one form.
- For ex, consider the operation of addition.
- If the operands are strings, then the operation would produce a third string by concatenation.
- If the operands are numbers, it will generate a sum.

10.Define dynamic binding

- It refers to the linking of a procedure call to the code to be executed in response to the call.
- Dynamic binging means that the code associated with a given procedure call is not known until the time of the call at run time.
- It is associated with polymorphism and inheritance

11. What r the benefits of OOP

- Through inheritance, we can eliminate redundant code and extend the use of existing classes.
- It saves the program development time and higher productivity.
- The use of data hiding helps the programmer to build secure programs that cannot be invaded by code in other parts of the program
- It is possible to have multiple instances of an objects to co-exist without any interference.
- It is possible to map objects in the program domain to those objects in the program.
- It is easy to partition the work in a project based on objects.
- It can be upgraded from small to large systems.
- It makes the interface descriptions with external systems much simpler by using message passing techniques.
- Software complexity can be easily managed.

12. What are the 2 types of OOP?

1. Object-Based Programming Languages
2. Object-Oriented Programming Languages

<u>Object- Based Programming Languages</u>

- It is the style of programming the primarily supports encapsulation and object identity.
- Major features are:

1. Data encapsulation

2. Data hiding and access mechanisms
3. Automatic initialization and clear-up of objects
4. Operator Overloading

- It doesn't support inheritance and dynamic binding
- Ex: Ada

Object-Oriented Programming Languages

- It combines Object- based programming features along with 2 additional features, inheritance and dynamic binding
- Ex: C++, Smalltalk

13. What are the applications of OOP

- Real time systems
- Simulation and modeling
- Object oriented data bases
- Hypertext, hypermedia and expertext
- AI and expert systems
- Neural networks and parallel programming
- Decision support and office automation systems
- CIM\CAM\CAD systems

14. What is C++?

- C++ is an object-oriented programming language
- Also known as C with classes , combination of C and simula 67
- Invented by Bjarne Stroustrup at AT&T Bell Lab in New Jersey, USA.
- It is an extension of C with a major addition of the class construct feature of simula 67.
- The 3 most important features That C++ adds on to C are classes, function overloading and operator overloading.
- It allows the programmer to build large programs with clarity, extensibility and ease of maintenance, incorporating the sprit and efficiency of C

15. What are the applications of C++?

- It is a versatile language for handling very large programs
- It is suitable for virtually any programming task including development of editors, compilers, databases, communication systems and any complex real life application systems
- It allows us to create hierarchy-related objects, so we can build special object-oriented libraries which can be used later by many programmers
- While C++ is able to map the real-world problem properly, the C part of C++ gives the language the ability to get close to the machine – level details
- C++ programs are easily maintainable and expandable

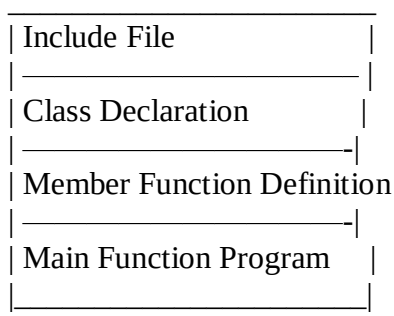
16. Why we need the preprocessor directive #include?

- This directive causes the preprocessor to add the content of the iostream file to the program. It contains declarations for the identifier cout and the operator <<.
- It should be included at the beginning of all programs that use i/o statements.

17. How does a main() function in C++ differ from main() in C?

In C++, main() returns an integer type value to the operating system. Therefore , every main() in C++ should end with a return (0) statement. Otherwise a warning or an error occurs. Since main() returns an integer type value, return type for main() is explicitly specified as int.
In C, it does not specify any return value for the main() function. So there is no need for explicitly use the return statement.

18. Describe major parts of a C++ program.



The class declarations are placed in a header file and the definitions of member functions go into another file. The main program that uses the class is placed in a third file which " includes" the previous two files as well as any other files required.

19. Describe I/O operator.

Input Operator

The statement **cin>> num;**

is an input statement and causes the program to wait for the user to type in a number. The operator >> is known as extraction or get from operator. It takes the value from the keyboard and assigns it to the variable on its right.

Output Operator

The statement **Cout<< " the numbers";**

uses the cout identifier that represents the standard output stream (screen) in C++. The operator <

20. Describe tokens

The smallest individual units in a program are known as tokens. C++ has the following tokens:

1. Keywords
2. Identifiers
3. Constants
4. Strings
5. Operators.

21. Classifications of Data Types.

1. User defined Type
    a. Structure
    b. Union
    c. Class
    d. Enumeration
2. Built in type
    a. integral type

i. int
ii. char

b. void

c. floating type
i. float
ii. double

d. Derived type
i. array
ii. function
iii. pointer
iv. reference

22.Describe with example the uses of enumeration data type

(i) It provides a way for attaching names to numbers.

(ii) The enum keyword gives the list of words by assigning them values 0,1,2 and so on. The Syntax is **enum identifier { enumerated constants };**

Ex: enum shape{ circle, square, triangle};

enum color { red, blue, green };

colur background; /* background is of type color */

(iii) Here each enumerated data type retains its own separate type. Ex:

color background = blue;

color background = (color) 7;

(iv) By default, the enumerators are assigned integer values starting with 0 for the first enumerator. But this can be over-ride. For ex,

enum color { red, blue=4, green =8};

[Write one example c++ program]

23. Explain Symbolic Constants

There are two ways of creating symbolic constants

(i) Using the qualifier Constants

=> Any value declared as const can't be changed.

=> ex: const int size =10;

char name[size];

(ii) Defining a set of integer constants using enum keyword

For ex, enum { X, Y, Z];

This defines X, Y,Z as integer constants with values 0,1,2 respectively.

This can be also assigned explicitly. For ex,

enum { X=100, Y=200, Z=300 };

24. What do u mean by dynamic initialization of a variables?

Initialization of variable at run time is known as dynamic initialization of a variables. That is the variables can be initialized at run time using expression at the place of declaration.

Ex:

……

……

int n = strlen (string);

……..
for(int i=0;i<5;i++)
{
……
……
}
By using this, we can create exactly the type of object needed, using the information that is known only at the run time.

25. What are the operators available in C++?

- Arithmetic operator
- Relational Operator
- Assignment Operator
- Logical Operator
- Increment/ decrement Operator
- Conditional operator
- Bitwise operator
- Special operator
- Scope resolution operator
- Pointer to member declaration
- Pointer to member operator
- Memory release operator
- Line feed operator
- Filed width operator

26.Explain about Type Casting Operator

This is used for conversion of variables or expression explicitly
Syntax:
(type – name) expression
or
Type –name (expression)
Ex: avg= sum/ float(i);

Alternatively, we can use typedef to create an identifier of the required type and use it in the functional notation
typedef int * int_pt;
p= int_pt(q);

27.What are the types of expression?

- Constant expression
- Integral expression
- Float expression
- Pointer Expression
- Relational Expression
- Logical expression
- Bitwise expression

28. What is meant by operator overloading?

Overloading means assigning different meaning to an operation, depending on the context. That is it is used to assign multiple meanings to operators.
Ex:
The operator * when applied to a pointer variable, gives the value pointed to by the pointer. But it is used for multiplying two numbers.

29. What are the types of control strucuture

- sequence
- selection
- iteration

30.What are the advantages of new operator

- It automatically computes the size of the data object. So there is no need to use sizeof operator
- It automatically returns the correct pointer type, so that here is no need to use a type cast.
- It is possible to initialize the object while creating the memory space
- Like any other operator, operator new and delete can be overloaded.

10 . what is the use of a break statement ?

A break construct terminates the execution of loop and the control is transferred to the statement immediately following the loop. The term break refers to the act of breaking out of a block of code.

11.What is the use of a continue statement ?

The continue statement skips the remainder of the current iteration initiates the execution of the next iteration.

Unit-II

1. **What is a class ?**
   It is an extension to the structure data type. A class can have both variables and functions as members

2. **What is the difference between structure and a class ?**
   The only difference between a structure and a class in C++ is that , by default , the members of a class are private, while , by default the members of a structure are public.

3. **What is the specification for a class ?**
   - Class declaration
   - Class function definitions
   -

4. **What are data members and member functions?**
   The variables declared inside the class are known as data members and the functions are known as member functions. The data members are usually private and member functions as public.

5. **Give a simple class example.**

```
class item
{
int number;
float cost;
```

```
public:
void getdata(int a, float b);
void putdata(void);
};
```
Here class name is item
Data : number, cost
Functions: getdata(), putdata()

6. **What are objects ?**

The class variables are called objects. With objects we can access the public members using dot operator

7. **How is a member function of a class is defined?**

It can be defined either inside or outside the class

8. **What are the characteristics of member functions ?**

- Several different classes can use the same function name. the membership label will resolve their scope
- Member functions can access the private data of the class. Anon member function cannot do so
- A member function can call another function directly ,without using dot operator

9. **When a function is defined inside a class ?**

- it is treated as a inline function
- only small functions are defined inside the class definition

10. **What is nesting of member functions ?**

Amember function can be called by using its name inside another member function of the same class, is known as member function

11. **How the space is allocated for the objects?**

The memory space is allocated when they are declared . space for the member variables is allocated separately for each object, but no separate space is allocated for the member functions

12. **When do we declare a member of a class static ?**

When it is used to maintain values common to the entire class. The static member variables defined outside the class

13. **What is a friend function?**

The functions that are declared with the keyword friend are known as friend functions. A function can be declared as a friend in any number of classes, it has full access rights to the private members of the class.

14. **What are the properties of a friend function ?**

· A friend function is not in the scope of the class to which it has been declared as friend.
· It can be invoked like a normal function without the help of any object.
· Unlike member functions it cannot access the member names directly and has to use an object name and dot membership with each member name.

15. **What is the difference between friend function and member function ?**

The only difference between a friend function and member function is that, the friend function requires the argument to be explicitly passed to the function and processes them explicitly, whereas, the member function considers the first argument implicitly.

16. **What is an inline function ?**

Inline functions are those whose function body is inserted in place of the function call statement during the compilation process. With the inline code the program will not incur any context switching overhead.

17. **What is a recursive function ?**

A function that contains a function call to itself or a function call to a second function which eventually calls the first function is known as recursive functions.

18. **What are the special characteristics of friend function ?**

1. Can be invoked like a normal function, with the help of the object
2. It has the objects as arguments
3. It is not in the scope of the class to which is has been declared has friend

19. **What is const member function ?**

If a member function does not alter any data in the class, then we declare it as const member function. The keyword const is appended to the function prototype.

20. **What is a constructor ?**

It is a special member function whose task is to initialize the objects of its class. It is special because its name is the same name as the class name.

21. **How do we invoke constructor function?**

It is invoked whenever an object of an associated class is created. It is called constructor because it constructs the values of data members of the class.

22. **List some special properties of constructor functions**.

1. They should be declared in the public section
2. They are invoked automatically when the objects are created
3. They do not have return types, therefore they cannot return values
4. They cannot be inherited
5. They can have default arguments
6. Cannot refer to addresses

23. **What is parameterized constructor ?**

It is nothing but passing arguments to the constructor function when the objects are created. The constructor can take arguments are called parameterized constructor.

24. **What is copy constructor ?**

The constructor that creates a new class object from an existing object of the same class.

25. **What is dynamic initialization of objects ?**

The initial value of an object provided at the run time. The advantage is that we can provide various initialization formats ,using overloaded constructors.

26. **What is dynamic constructor ?**

Allocation of memory to objects at the time of their construction is known as dynamic construction of objects. The memory is allocated with the help of new operator.

27. **What is a destructor?**

It is used to destroy the objects that have been created by a constructor, when they no longer required.

Unit-III

1.What is operator overloading?

Operator overloading is a compile-time polymorphism in which the operator is overloaded to provide the special meaning to the user-defined data type. Operator overloading is used to overload or redefines most of the operators available in C++. For example, C++ provides the ability to add the variables of the user-defined data type that is applied to the built-in data types.

2.List the Operator that cannot be overloaded.

Operator that cannot be overloaded are as follows:

- Scope resolution operator (::)
- Size operator (sizeof)
- Class member access operator (.)
- Pointer to member operator (.*)
- Ternary (or) Conditional operator(?:)

3. What is the purpose of using operator function? Write its syntax.

To define an additional task to an operator, we must specify what it means in relation to the class to which the operator is applied. This is done by Operator function , which describes the task. Operator functions are either member functions or friend functions. The general form is

```
return type classname :: operator op(arglist )
{
function body
}
```

where *return type* is the type of value returned by specified operation.

Op-operator being overloaded. The *op* is preceded by a keyword operator. operator op is the function name.

4. How will you overload Unary & Binary operator using member functions? When unary operators are overloaded using member functions it takes no explicit arguments and return no explicit values.

When binary operators are overloaded using member functions, it takes one explicit argument. Also the left hand side operand must be an object of the relevant class.

5. How will you overload Unary and Binary operator using Friend functions? When unary operators are overloaded using friend function, it takes one reference argument (object of the relevant

class)

When binary operators are overloaded using friend function, it takes two explicit arguments.

6. How an overloaded operator can be invoked using Friend functions?

In case of unary operators, overloaded operator can be invoked as

Operator op (x);

In case of binary operators, overloaded operator can be invoked as

Operator op (x , y)

7. **List out the operators that cannot be overloaded using Friend function.**

Operator that cannot be overloaded by friend function are as follows:

- Assignment operator =
- Function call operator ( )
- Subscripting operator [ ]
- Class member access operator ->

5. What are process of operator overloading?

1. Create a class that defines the data type that is to be used in the overloading operation.
2. Declare the operator function **operator** op() in the public part of the class. It may be either a member function or a **friend** function.
3. Define the operator function to implement the required operations.

6. What are the rules for overloading operators?

- Existing operators can only be overloaded, but the new operators cannot be overloaded.
- The overloaded operator contains atleast one operand of the user-defined data type.
- We cannot use friend function to overload certain operators. However, the member function can be used to overload those operators.
- When unary operators are overloaded through a member function take no explicit arguments, but, if they are overloaded by a friend function, takes one argument.
- When binary operators are overloaded through a member function takes one explicit argument, and if they are overloaded through a friend function takes two explicit arguments.

7. Explain basic to class type conversion with an example.

Conversion from basic data type to class type can be done in destination class. Using constructors does it. Constructor takes a single argument whose type is to be converted.

Eg: Converting int type to class type class time

```
{
int hrs,mins;
public:
…….
Time ( int t) //constructor
{
hours= t/60 ; //t in minutes mins =t % 60;
}
};
```

Constructor will be called automatically while creating objects so that this conversion is done

automatically.

8. Explain class to basic type conversion with an example.

Using Type Casting operator, conversion from class to basic type conversion can be done. It is done in the source class itself.

Eg: vector : : operator double( )
{
double sum=0;
for(int I=0;I<size;I++) sum=sum+v[i] *u[i] ; return sqrt (sum) ;
}

This function converts a vector to the corresponding scalar magnitude.

9. What is casting operator function?

A Cast operator is an unary operator which forces one data type to be converted into another data type.

```
operator typename()
{
    …….
    …….(function statements)
    …….
}
```

10. What are the rules for casting operator function?

- It must be a class member.
- It must not specify a return type.
- It must not have any arguments.

11.What is meant by reusability

Reusability is a feature which is supported in object-oriented programming. This allows the reuse of existing classes without redefinition.

12.Define Inheritance

- In object-oriented programming, **inheritance** is a way to form new classes (instances of which are called objects) using classes that have already been defined.
- The former, known as **derived classes**, take over (or **inherit**) attributes and behavior of the latter, which are referred to as **base classes**.
- It is intended to help reuse of existing code with little or no modification.
- **Inheritance** is also called **generalization**, because the **is-a** relationships capture a hierarchal relationship between classes of objects

13. **What are the applications of Inheritance**

There are many different aspects to inheritance. Different uses focus on different properties, such as the external behavior of objects, internal structure of the object, structure of the inheritance hierarchy, or software engineering properties of inheritance.

(i)Specialization

One common reason to use inheritance is to create specializations of existing classes or objects. This is often called *subtyping* when applied to classes. In specialization, the new class or object has data

or behavior aspects that are not part of the inherited class.

Another form of specialization occurs when an inherited class specifies that it has a particular behavior but does not actually implement the behavior. Each non-abstract, concrete class which inherits from that abstract class must provide an implementation of that behavior. This providing of actual behavior by a subclass is sometimes known as *implementation* or *reification.*

(ii)Overriding

Many object-oriented programming languages permit a class or object to replace the implementation of an aspect—typically a behavior—that it has inherited. This process is usually called *overriding*.

(iii)Extension

Another reason to use inheritance is to provide additional data or behavior features. This practice is sometimes called *extension* or *subclassing*.

Extension is often used when incorporating the new features into the inherited class is either not possible or not appropriate.

(iv)Code re-use

One of the earliest motivations for using inheritance was to allow a new class to re-use code which already existed in another class. This practice is usually called *implementation inheritance*.

14.What are the Constraints of inheritance-based design?

- **Singleness**: using single inheritance, a subclass can inherit from only one superclass. Continuing the example given above, Person can be either a Student or an Employee, but not both. Using multiple inheritance partially solves this problem, as a Student,Employee class can be defined that inherits from both Student and Employee. However, it can still inherit from each superclass only once; this scheme does not support cases in which a student has two jobs or attends two institutions.
- **Staticness**: the inheritance hierarchy of an object is fixed at instantiation when the object's type is selected and does not change with time. For example, the inheritance graph does not allow a Student object to become aEmployee object while retaining the state of its Person superclass.
- **Visibility**: whenever client code has access to an object, it generally has access to all the object's superclass data. Even if the superclass is not a public one, the client can still cast the object to its superclass type. For example, there is no way to give a function a pointer to a Student's grade point average and transcript without also giving that function access to all of the personal data stored in the student's Person superclass.

15. What are the types of inheritance?

1. Single inheritance: A derived class with only one base class is called single inheritance
2. Multiple inheritance: A class can inherit properties from more than one class which is known as multiple inheritance
3. Multilevel inheritance: A class can be derived from another derived class which is known as multilevel inheritance.
4. Hierarchical inheritance: When the properties of one class are inherited by more than one class, it is called hierarchical inheritance.

16. How to define derived classes?

A derived class can be defined by specifying its relational ship with the base class in addition to its own details.

The syntax is:
class derived-class-name : visibility-mode base-class-name
{
……
…….
};
Here the visibility mode is optional and if present, may be either private or public. The default mode is private.
When a base class is privately inherited by a derived class, public members of the base class become private members of the derived class and therefore the public members of the base class can only be accessed by the member functions of the derived class.
When the base class is publicly inherited, public members of the base class become public members of the derived class and therefore they are accessible to the objects of the derived class.

17. What is inherited from the base class?
In principle, a derived class inherits every member of a base class except:

- its constructor and its destructor
- its operator=() members
- its friends

Although the constructors and destructors of the base class are not inherited themselves, its default constructor (i.e., its constructor with no parameters) and its destructor are always called when a new object of a derived class is created or destroyed.

18.Define virtual base class

- A base class that has been qualified as virtual in the inheritance definition.
- In multiple inheritance, a derived class can inherit the members of a base class via two or more inheritance paths.
- If the base class is not virtual, the derived class will inherit more than one copy of the members of the base class.

19. What is meant by subclass and superclass
Subclass: a class which has link to a more general class
Superclass: a class which has one or more members which are classes themselves.

20. Define abstract class
Abstract class is one that is not used to create objects. An abstract class is designed only to act as a base class. It is a design concept in program development and provides a base upon which other classes may be built.

21.What are the forms of inheritance

- Single inheritance
- Multiple Inheritance
- Multilevel Inheritance
- Hierarchal Inheritance
- Hybrid Inheritance

Unit-IV

1. What is meant by this pointer?

- C++ uses the unique keyword called **this** to represent an object that invokes a member function.
- **this** is a pointer that points to the object for which this function was called.
- This unique pointer is automatically passed to a member function when it is called. The pointer this acts as an implicit argument to all the member functions.
- One important application of the **this** pointer this is to return the object it points to.

2. What is meant by pointer and null pointer?

Pointer = pointer is a data type that holds the address of a location in memory.

Null Pointer = is a pointer that does not point to any data object. In C++, the null pointer can be represented by the constant 0.

3. **What is this pointer?**

this pointer refers to the current object of the class and passes it as a parameter to another method. *'this pointer*' is passed as a hidden argument to all non-static member function calls.

4. Define virtual function

- It is a function qualified by the virtual keyword. When a virtual function is called via a pointer, the class of the object pointed to determines which function definition will be used.
- Virtual functions implement polymorphism, whereby objects belonging to different classes can respond to the same message in different ways.

5. What is meant by pure virtual function

A pure virtual function is a virtual function in C++ for which we need not to write any function definition and only we have to declare it. It is declared by assigning 0 in the declaration.

```
virtual void s() = 0; // Pure Virtual Function
```

6. What are the rules for virtual function?

The virtual function must be members of some function.

- They can't be static members
- They are accessed by using object pointers.
- A virtual function can be a friend of another class
- A virtual function in a base class must be defined, even though it may not be used.
- The prototypes of the base class version of a virtual function and all the derived class versions must be identical. If two functions with the same name have different prototypes, C++ considers them as overloaded functions, and the virtual function mechanism is ignored.
- We cannot have virtual constructors, but we can have virtual destructors.
- While a base pointer can point to any type of the derived object, the reverse is not true.
- When a base pointer to a derived class, incrementing or decrementing it will not make it to point to the next object of the derived class. It is incremented or decremented only relative to its base type. Therefore we should not use this method to move the pointer to the next object.
- If a virtual function is defined in the base class, it need not be necessarily redefined in the derived

class. In such cases, calls will invoke the base function.

Unit-V

1. What are C++ Streams?
The C++ language offers a mechanism, which permits the creation of an extensible and consistent input-output system in the form of streams library. It is a collection of classes and objects which can be used to build a powerful system or it can be modified and extended to handle user defined data types

2. List the predefined console streams.a) cin – standard input
b)cout – standard output
c) cerr – standard error output
d)clog – fully buffered version of cerr

3. Give the usage of ios class.
The ios class provides operations common to both input and output. It contains a pointer to a buffer object. It has constants and member functions that are useful in handling formatted I/O operations.
Following are the derived classes of ios class,
a) istream – input stream
b)ostream – output stream
c)iostream – input-output stream

4. What are the types of formatted console i/o operations?
a) ios stream class member functions and class
b) standard manipulators
c) user defined manipulators

5. Give the flag value and bit field for (a) Left justified output and (b) Decimalconversion.
Flag valueBit field
(a) Left justified output ios::left ios::adjustfield
(b) Decimal conversion ios::dec ios::basefield

6. What are the types of manipulators? Give example.
Two types of manipulators are available in C++.
a) **Parameterized manipulators**
Eg: setw(int width) – sets the field width
setprecision(int prec) – sets the floating point precision
b) **Non-parameterized manipulator**
Eg: dec – sets the conversion base to 10
Endl – outputs a new line and flushes stream

7. What is a custom manipulator? Give its syntax.
Designing of customized manipulators to control the appearance of the output is referred as custom manipulator.
Syntax:

```
ostream& manipulator(ostream& output, arguments_if_any)
{//manipulator code
return output;
}
```

8. Write a note on File.

A file is a collection of related information normally representing programs, both source and object forms and data. Data may be numeric, alphabetic or alphanumeric. A file can also be defined as a sequence of bits, bytes, lines or records whose meaning is defined by the programmer. Operations such as create, open, read, write and close are performed on files

9. Define ifstream&fstream.

ifstream:

It is used for handling input files. It contains open() with default input mode and inherits get(), getline(), read(), seekg(), tellg() functions from istream.

fstream:

Used for handling files on which both i/o operations can be performed. It supports simultaneous i/o operations. It contains open() with default input mode and inherits all the functions from istream and ostream classes through iostream.

10. Give the prototypes of file stream class constructors.

a) ifstream class constructor

ifstream(const char *path, int mode=ios::in, int prot=filebuf::openprot);

b) ofstream class constructor

ofstream(const char *path, int mode=ios::out int prot=filebuf::openprot);

c) fstream class constructor

fstream(const char *path, int mode=ios::in/ios::out, int prot=filebuf::openprot);

11. List any four file modes and their purpose

a) ios::in – open for reading

b) ios::ate – seek to the end of file at opening time

c) ios::nocreate – open fails if file does not exist

d) ios::binary – opens a binary file

12. List the file pointer control functions.

a) seekg() – moves get file pointer to a specific location

b) seekp() – moves put file pointer to a specific location

c) tellg() – returns the current position of the get pointer

d) tellp() – returns the current position of the put pointer

13. What are the types of file accessing?

a) Sequential access

This type of file is to be accessed sequentially that is to access a particular data all the preceding data items have to be read and discarded.

b) Random access

This type of file allows access to the specific data directly with out accessing its preceding data items

14. What do you mean by sequential access ?

A sequential file has to be accessed sequentially ; to access the particular data in the file all the preceding data items have to be read and discarded. For example a file on a tape must be accessed sequentially.

15. What do you mean by random access ?

A random file allows access to the specific data without the need for accessing its preceding data items. However, it can be accessed sequentially . For example, a file on a hard disk or floppy disk can be accessed either sequentially or randomly.

16. Give any two error handling functions and their purpose

a) eof() – TRUE(nonzero) if eof encountered while reading; otherwise FALSE(zero)
b) rdstate() – returns the status state data member of the class ios

17. Define fault avoidance and fault tolerance

Fault avoidance:

It deals with the prevention of fault occurrence by construction. It emphasizes on techniques to be applied during system development that ensures the system satisfies all reliability criteria

Fault tolerance:

This deals with the method of providing services complying with the specification in spite of false occurring by redundancy.

18. What is exception handling? How it is classified?

The error handling mechanism of C++ is referred to as exception handling. Exception refers to some unexpected condition in a program. It is classified as synchronous and asynchronous exception.

Synchronous exception :

The exceptions, which occur during the program execution, due to some fault in input data, within the program, is known as Synchronous exception.

Asynchronous exception :

The exceptions caused by events or a fault unrelated to the program and beyond the control of the program is known as asynchronous exception.

19. Define the exception handling constructs.

a) try

This keyword defines a boundary within which an exception can occur. A block of code in which an exception may occur must be prefixed by this keyword.

b) throw

Throw is used to raise an exception when an error is generated in the computation. It initializes a temporary object to be used in throw.

c) catch

This keyword represents exception handler. It must be compulsorily used immediately after the statements marked by try keyword. It can also occur immediately after catch keyword. Each handler will only evaluate an exception that matches or can be converted to the type specified in the argument list

20. What are the tasks to be performed by error handling code?

a) Hit the exception – detect the problem causing exception

b) Throw the exception – inform that an error has occurred
c) Catch the exception – receive the error information
d) Handle the exceptions – take corrective actions

21. List the functions for handling uncaught exceptions.
a) **terminate()** – it is invoked when an exception is raised and the handler is not found.
b) **set_terminate()** – allows the user to install a function that defines the program's actions to be taken to terminate the program when a handler for the exception cannot be found
c) **unexpected()** – this function is called when a function throws an exception not listed in its exception specification
d) **set_unexpected()** – it allows the user to install a function that defines the program's actions to be taken when a function throws an exception not listed in its exception specification

22. How fault tolerant s/w design techniques are classified?
a) **N-version programming** – in this technique, N-programmers develop N algorithms for the same problem with out interacting with each other. All these algorithms are executed simultaneously on a multiprocessor system and the majority solution is taken as the correct answer.
b) **Recovery block** – this structure represents the dynamic redundancy approach to s/w fault tolerance. It consists of
(i) Primary routine– executes critical s/w function,
(ii) acceptance test – tests the output of primary routine after each execution and
(iii) alternate routine – performs the same function as primary routine but is invoked by an acceptance test after reduction of a fault.

23**.What is a template ?**
Templates support generic programming , which allows to develop reusable software components such as functions, classes etc., supporting different data types in a single framework

24.What is function template ?
The templates declared for functions are called function templates. They perform appropriate operations depending on the data type of the parameters passed to them.

25.What is a class template ?
The templates declared for classes are called function templates. They perform appropriate operations depending on the data type of the parameters passed to them.

26.Define runtime exceptions
Runtime exceptions are automatically defined for the programs that we write and include things such as division by zero and invalid array indexing.

27.Give the general form of throw . throw throwableinstance;
throwable instance must be an object of type throwble or a subclass of throwable.

Important Questions

Unit-I

Part-A

1.Define Object and class.
2.Compare procedural language and object oriented language
3.What is encapsulation?
4.What is meant by message passing?
5.what are the advantages of Object oriented language?
6.what are the difference between structure and class.
7.List the types of polymorphism.
8.Compare private and protected access specifier
9.Define friend function and explain its characteristics.
10.What do we use the protected visibility specifier to a class member?
11.Define this pointer in c++.
12.What is function Overloading
13.List out the operators which cannot be overloaded and cannot be overloaded using friend function.
14.Differentiate member function and friend function in overloading.
15.What is class template?
16.Define an abstract class.
17.What is template function?
18.Write the advantages of multiple inheritance?
19.How do we declare a member of a class 'static'?
20.Define nested classes.
21.Define volatile Function.
22.Define Local class.
23.What is meant by Pointer?
24.What is use of Access specifier?
25.What is meant by Abstraction?

Part-B

1.a.what are advantage of using OOPS concepts.
b.Explain the nested classes with example.
2.What are access specifier?How are they used to protect data in C++?
3.Explain a multilevel,Multiple and Multipath inheritance.
4.Explain the friend class and Friend function with examples?
5.a.Explain Default Arguments.
b.Explain about static and const Member Functions.
6.Explain in detail about pointers and Objects in C++?

Unit-II

Part-A

1.What is a Constructor?
2.Define Copy constructor.
3.Define Destructor.

4.What is Operator Overloading?
5.Define default constructor.
6.What is the difference between Explicit and Implicit casting.?
7.What is meant by Assignment Operator.
8.What is the use of defining Constructor in the program?
9.Differnce between constructor and destructor?
10.What is the use of Constructor with Dynamic allocation.
11.Define Assignment Operator.
12.Define Explicit constructor.
13.Write the syntax for unary operator Overloading?
14.Write the Syntax for Binary operator Overloading?
15.Write the syntax for invoking overloaded unary and binary operator?

Part-B

1.Explain in detail about Operator Oveloading?
2.Explain about Overloading in the assignment Operator?
3. Describe briefly about overloading through friend functions.
4.Write a program to implement Constructor ,Copy constructor and Destructor?
5.a.What is the use of Explicit Constructor.
b. Explain in detailed about Constructor in Dynamic allocation.
6.Write a program for overloading new and delete operators.

Unit-III

Part-A

1.Define Template.
2.What is meant by Function Template.
3.Define Exception Handling.
4.What are the type of Error?
4.Define Terminate functions.
5.What is meant by Uncaught exception?
6.List some Exception.
7.Dfine try and catch.
8.What is use of rethrow?
9.Name some of the Standard Library Exceptions.
10.What is meant by multi-catch Exception.?
11.Write the syntax for Function template?
12.Define Exception Specification.
13.What is use of Template?

Part-B

1.Explain in detail about Exception Handling in C++.?
2.a.Write a program for Function Template?

b. Write a program for Class Template?
3.Describe in detailed about Unexpected functions and Uncaught exception.
4.Write a brief note on try-catch-throw Model?
5.a.What are the rules for Handling Exceptions Successfully?
b.Explain Exception in class template?

Unit –IV

Part-A

1.Define Inheritance.
2.What are the Access specifier used in Inheritance?
3.Define Multiple Inheritance.
4.Define Abstract Class.
5.Define Virtual Functions
6.Expand RTTIandDefirne RTTI.
7.Define typeid.
8.What is meant by cross casting?
9.Write the uses of virtual base class?
10.Write the syntax for dynamic cast?

Part-B

1.Explain in detailed about types of Inheritance?
2.Describe about RTTI?
3.Explain
a.Multiple Inheritance
b.Pure Virtual Function.
4.Define
(i)Down casting
(ii)Cross casting
(iii)Composite Objects Runtime Polymorphism.
5.Explain in detailed about Virtual and pure virtual Function?

Unit-V

Part-A

1.Define Stream.
2.What is meant by namespaces?
3.Define setf().
4.How will u display trailing Zeros in the output?
5.Define file.
6.What is meant STL?
7.Define Object searilization.
8.Name some file modes?
8.Define string objects.
9.What is meant by ANSI?
10.Define Manipulators.

11.what are the types of manipulators?
12.Define ios::showbase.
13.what is meant by ifstream?
14.Write the syntax for file open and close?
15.what is need of STL?

Part-B

1.Write in detailed about formatted I/O?
2,Explain about Standard template library?
3.Explain
a.Objectseralization
b.std namespaces
4.Explain in detailed about Manipulators?
5.Explain the hierarchy of File stream classes?
6.Write about Random Access to a file?