CS2203 - Object Oriented Programming

Branch: B.E Computer Science and Engineering
Year: II Semester: III

Prepared By,

L. Maria Michael Visuwasam
Assistant Professor, CSE

Unit - 1

1. State the characteristics of Procedure Oriented Programming.

- ✓ Emphasis is on doing things(Algorithms)
- ✓ Large programs are divided into smaller programs known as functions
- ✓ Most of the function share global data
- ✓ Data move openly around the system from function to function
- ✓ Functions transform data from one form to another
- ✓ Employs top-down approach in program design.

2. What are the features of Object Oriented Programming?

- ✓ Programs are divided into objects
- ✓ Emphasis is on data rather than procedure
- ✓ Data Structures are designed such that they characterize the objects
- ✓ Functions that operate on the data of an object are tied together
- ✓ Data is hidden and cannot be accessed by external functions
- ✓ Objects may communicate with each other through functions
- ✓ New data and functions can easily be added whenever necessary
- ✓ Follows bottom-up approach.

3. Distinguish between procedure oriented programming and object oriented programming.

- ✓ Emphasis is on doing things(Algorithms)
- ✓ Large programs are divided into smaller programs known as functions
- ✓ Function share global data
- ✓ Data move openly around the system from function to function

- ✓ Employs top-down approach in program design.
- ✓ Emphasis is on data rather than procedure
- ✓ Programs are divided into objects
- ✓ Functions that operate on the data of an object are tied together
- ✓ Data is hidden and cannot be accessed by external functions
- ✓ Follows bottom-up approach

4. Define Object Oriented Programming(OOP).

Object Oriented programming 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.

5. List out the basic concepts of Object Oriented Programming.

- ✓ Objects
- ✓ Classes
- ✓ Data Abstraction and Encapsulation
- ✓ Inheritance
- ✓ Polymorphism
- ✓ Dynamic Binding
- ✓ Message Passing

6. Define Objects

Objects are the basic run-time entities in an object oriented system. They are instance of class. They may represent a person, a place ,a bank account etc that a program has to handle. They may also represent user-define data.

7. Define Class.

Class is a collection of objects of similar data types. Class is a user-defined data type. The entire set of data and code of an object can be made a user define type through a class.

8. Define Encapsulation and Data Hiding.

The wrapping up of data and functions into a single unit is known as data encapsulation. Here the data is not accessible to the outside world. The insulation of data from direct access by the program is called data hiding or information hiding.

9. Define Abstraction.

Abstraction refers to the act of representing the essential features without including the

backgrounf details or explanations.

10. Define data members and member function.

The attributes in the objects are known as data members because they hold the information. The functions that operate on these data are known as methods or member functions.

11. State Inheritance.

Inheritance is the process by which objects of one class acquire the properties of objects of another class. It supports the concept of hierarchiacal classification and provides the idea of reusability. The class which is inherited is known as base or super class and class with is newly derived is known as the derived or sub class.

12. State Polymorphism.

Polymorphism is an important concept of OOPs. Polymorphism means one name, multiple forms. It is the ability of a function or operator to take more than one form at different instances.

13. List and define the two types of Polymorphism.

Operator Overloading – the process of making an operator to exhibit different behaviors at different instances.

Function Overloading – Using a single function name to perform different types of tasks. The same function name can be used to handle different number and different types of arguments.

14. State dynamic Binding.

Binding refers to the linking of procedure call to the code to be executed in response to the call. Dynamic Binding or late binding means that the code associated with a given procedure call is known only at the run-time.

15. Define Message Passing.

Objects communicate between each other by sending and receiving information known as messages. A message to an object is a request for execution of a procedure. Message passing involves specifying the name of the object the name of the function and the information to be sent.

16. List out some of the benefits of OOP.

- ✓ Eliminate redundant code
- ✓ Saves development time and leads to higher productivity
- ✓ Helps to build secure programs

- ✓ Easy to partition work
- ✓ Small programs can be easily upgraded to large programs
- ✓ Software complexity can easily be managed.

17. List out the applications of OOP.

- ✓ Real time systems
- ✓ Simulation and modeling
- ✓ Object oriented databases
- ✓ Hypertext, hypermedia and expertext
- ✓ AI and expert systems
- ✓ Neural networks and parallel programming.

18. Define C++

C++ is an object oriented programming language developed by Bjarne Stroustrup. It is a super set of C. Initially it was known as "C with Classes". It is a versatile language for handling large programs.

19. What are the input and output operators used in C++?

The identifier cin is used for input operation. The input operator used is >>, which is known as the extraction or get from operator The Syntax is cin>>n1.

The identifier cout is used for output operation. The operator used is <<, which is known as the insertion or put to operator The Syntax is cout<<"Welcome".

20. List out the four basic sections in a typical C++ program or Structure of C++ Program.

- ✓ Include files.
- ✓ Class declaration
- ✓ Member function definition.
- ✓ Main function Program.

21. What is meant by Default Argument

C++ allows us to assign degault values to the function parameters when the function is declared. In such a case we can call a function without specifying all its arguments. The defaults are always added from right to left.

Eg. Val=amount(900,5); // Function Call

Float amount(float pncl, int period ,float rate=0.15) //Function Definition

22. Define Function Overloading.

C++ allows function overloading. That is, we can have more than one function with the same name in our program. The compiler matches the function call with the exact function code by checking the number and type of the arguments.

23. What is const and volatile function

The qualifier const tells the compiler that the function should not modify the argument. The compiler generate an error when this condition is violated

The qualifier volatile tells the compiler that the function should modify the argument.

- ✓ **Eg:** int strlen(**const** char *p);
- ✓ int strlen(**volatile** char *p);

24. What are all characteristics of friend function?

- ✓ It cannot access the member names directly and uses the dot operator.
- ✓ It is not in the scope of a class in which it is declared as friend.
- ✓ It can be invoked without an object.
- ✓ It cannot be called using the object of that class.
- ✓ It has the objects as arguments.
- ✓ It can be declared as either public or private.

25. What is meant by static function?

Like static member variable we can also have static member function. A member function that is declared static has the following properties

- ✓ A static function can have access to only other static members(functions or variables) declared in the same class.
- ✓ A static member function can be called using the class name(instead of its objects)

26. Define static data member.

A data member of a class can be qualified as static. The properties of a static member variable are

- ✓ It is initialized to zero when the first object of its class is created. No other initialization is permitted.
- ✓ Only one copy of that member is created for the entire class and is shared by all the objects of that class, no matter how many objects are created.
- ✓ It is visible only within the class, but its lifetime is the entire program.

27. Define Local Classes.

Classes can be defined and used inside a function or a block. Such classes are called local classes.

```
Eg:     void test(int a)
{……..
class student     //local class
{ …….
```

```
    …….
};
}
```

27. Define Nested Classes.

A class can contain objects of other classes.. Such classes are called Nested classes.

28. What is abstract class

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

29. What is constant object?

One can create and use constant objects using **const** keyword before object declaration.

Eg: const matrix X(m,n); // object X is constant

Any attempt to modify the values of m and n will generate compile-time error.

30. Define token and identifiers.

The smallest individual units in a program are known as tokens. Various tokens are keywords, identifiers constants, strings and operators.

The identifiers refer to the names of variables, functions, arrays, classes etc created by the programmer.

31. State the advantages of default arguments.

- ✓ We can use default arguments to add new parameters to the existing function.
- ✓ Default arguments can be used to combine similar functions into one.

32. Why do we use default arguments?

The function assigns a default value to the parameters which do not have a matching argument in the function call. They are useful in situations where some arguments always have some

33. Define Class

A Class is a way to bind the data and its function together. It allows the data to be hidden from external use. The general form is

```
Class class-name
{
private:
```

var
declaration;
fn
declaration;
public
var
declaration;
fn
declaration;
};

34. List the access modes used within the class.

- ✓ Private: The class members are private by default. The members declared private are completely hidden from the outside world. They can be acceded from only within the class
- ✓ Public : The class members declared public can be accessed from any where.
- ✓ Protected: The class members declared protected can be access from within the class and also by the friend class.

35. Define Friend function.

An outside function can be made a friend to a class using the qualifier „friend". The function declaration should be preceded by the keyword friend. A friend function has full access to the private members of the class or the member function of friend class can directly access the private and protected data.

36. Define data members and member functions.

The attributes in the objects are known as data members because they hold the information. The functions that operate on these data are known as methods or member functions.

37. State the use of void in C++.

The two normal uses of void are

- ✓ To specify the return type of the function when it is not returning a value
- ✓ To indicate an empty argument list to a function

38. Define constant pointer and pointer to a constant.

The constant pointer concept does not allow us to modify the value initialized to the pointer.
e.g.) char * const ptr = "GOOD";
The pointer to a constant concept doesn"t allow us to modify the address of the pointer.
E.g.) int const * ptr = &n;

39. What are the two ways of creating symbolic constants?

- ✓ Using the qualifier const

- ✓ Defining a set of integer constants using enum keyword

40. Define reference variable. Give its syntax.

A reference variable provides an alias or alternate name for a previously defined variable. It must be initialized at the time of declaration. Its syntax is given by, data-type & reference-name = variable-name;

41. List out the new operators introduced in C++.

- **::** Scope resolution operator
  ::* Pointer to member
- declarator **->*** Pointer to
- member operator
  .* Pointer to member
- operator delete Memory
- release operator endl Line
- feed operator
- new Memory allocation operator
- setw Field width operator

42. What is the use of scope resolution operator?

A variable declared in an inner block cannot be accessed outside the block. To resolve this problem the scope resolution operator is used. It can be used to uncover a hidden variable. This operator allows access to the global version of the variable. It takes the form,

:: variable-name

43. List out the memory difreferencing operator.

::* To declare a pointer to the member of the class

->* To access a member using object name and a pointer to that member

.* To access a member using a pointer to the object and a pointer to that member

44. Define the 2 memory management operators.

- new Memory allocation operator

The new operator can be used to create objects of any data-type. It allocates sufficient memory to hold a data object of type data-type and returns the address of the object.

Its general form is, Pointer variable=new data-type;

- delete Memory release operator

When a data object is no longer needed it is destroyed to release the memory space for reuse.

The general form is, delete pointer variable;

45. List out the advantages of new operator over malloc ().

- ✓ It automatically computes the size of the data object.
- ✓ It automatically returns the correct pointer type.
- ✓ It is possible to initialize the objects while creating the memory space.
- ✓ It can be overloaded.

46. Define manipulators. What are the manipulators used in C++?

Manipulators are operators that are used to format the data display. Themanipulators used in C++ are

- ✓ endl – causes a linefeed to be inserted
- ✓ setw – provides a common field width for all the numbers and forces them to be printed right justified

47. What are the three types of special assignment expressions?

- ✓ Chained assignment e.g., x = y = 10;
- ✓ Embedded assignment e.g., x = (y = 50) + 10;
- ✓ Compound assignment e.g., x + = 10;

48. Define implicit conversion.

Whenever data types are mixed in a expression, C++ performs the conversions automatically. This process is known as implicit or automatic conversion. e.g., m = 5 + 2.75;

49. Define integral widening conversion.

Whenever a char or short int appears in an expression, it is converted to an int. This is called integral widening conversion.

50. What are the control structures used in C++?

- ✓ Sequence structure (straight line)
- ✓ Selection structure (branching)
  - if – else (two way branch)
  - switch (multiple branch)
- ✓ Loop structure (iteration or repetition)
  - do – while (exit controlled)
  - while (entry controlled)
  - for (entry controlled)

51. Define Function Prototyping.

The function prototype describes the function interface to the compiler by giving details such as the number and type of arguments and type of return values. It is the declaration of a function in a program. It is in the following form,

- ✓ type function – name (argument – list);
- ✓ where argument – list -> types and names of arguments to be passed to the function

52. What are inline functions?

An inline function is a function that is expanded in line when it is invoked. Here, the compiler replaces the function call with the corresponding function code. The inline function is defined as,

```
inline function-header
{
function body
}
```

53. List out the conditions where inline expansion doesn't work.

- ✓ For functions returning values, if a loop, a switch, or a goto exists
- ✓ For functions not returning values, if a return statement exists
- ✓ If functions contain static variables
- ✓ If inline functions are recursive

54. Differentiate structured programming and object oriented programming.

Structured Programming	Object Oriented Programming
Emphasis on algorithm rather than data	Emphasis on data rather than algorithm
The problem is divided into functions	The problem is divided into objects
Data are declared as global and accessible to all functions without any restrictions.	Data is encapsulated with the associated functions and this capsule is called an object.
It has reduced data security and integrity since the data is available to all functions.	It offers higher data security and integrity.
It does not support software reusability	It supports inheritance which provides software reusability.
Structured programming language has a top-down approach.	Object oriented programming language has a bottom-up approach.
Message passing is possible by means of parameter passing between functions.	Message passing is possible by means of communication between the objects.

55. Difference between structure and class.

Structure	Class
The keyword struct is used to create a structure	The keyword class is used to create a class.
The data member by default has public access specifier.	By default, it has private access specifier.
Once structure is accessed, all members defined inside, whether needed or not, are available.	Only the needed class members can be available.
It is possible to modify structure member accidently.	It is not possible to modify the class members accidently.

The syntax to create a structure is,
struct student
{
int rollno;
char name[25];
};
	The syntax is,
	class student
	{
	int rollno;
	char name[25];
	};

56. Difference between object assignment and object initialization.

When an assignment is applied to an object, that is when the statement object1=object2 is invoked, the value of the first member of object 1 is copied to the first member of object 2 and so on. It is important to note that both the objects must be the same type and all members have similar size.

In the case of object initialization, that is when the statement, classname object1=object2 invoked, it copies the entire object into another at a time.