INTERVIEW QUESTIONS FOR OBJECT-ORIENTED PROGRAMMING

ASSIGNMENT -2

**20. Explain an Inline function ?**

**Answer.**  An inline function is one for which the compiler copies the code from the function definition directly into the code of the calling function rather than creating a separate set of instructions in memory. This eliminates call-linkage overhead and can expose significant optimization opportunities.

**21. Explain a virtual function ?**

**Answer.** A virtual function or virtual method in an OOPs language is a function or method used to override the behavior of the function in an inherited class with the same signature to achieve the polymorphism.

**22. Explain function overloading?**

**Answer.** Function overloading is a feature of object-oriented programming where two or more functions can have the same name but different parameters. When a function name is overloaded with different jobs it is called Function Overloading.

**23.Explain a friend function?**

**Answer.** In object-oriented programming, a friend function, that is a "friend" of a given class, is a function that is given the same access as methods to private and protected data. A friend function is declared by the class that is granting access, so friend functions are part of the class interface, like methods.

**24.Explain a base class, sub class, super class?**

**Answer.** A class that is derived from another class is called a subclass (also a derived class, extended class, or child class). The class from which the subclass is derived is called a superclass (also a base class or a parent class).

**25.Write in brief linking of base class, sub class and base object, sub object.**

**Answer..**: In Java, all non-static methods are based on the runtime type of the underlying object rather than the type of the reference that points to that object. Therefore, it doesn’t matter which type you use in the declaration of the object, the behavior will be the same.

There are two approaches to refer a subclass object. Both have some advantages/disadvantages over the other. The declaration affect is seen on methods that are visible at compile-time.

1.First approach (Referencing using Superclass reference): A reference variable of a superclass can be used to a refer any subclass object derived from that superclass. If the methods are present in SuperClass, but overridden by SubClass, it will be the overridden method that will be executed.

2.Second approach (Referencing using subclass reference) : A subclass reference can be used to refer its object.

**26.Explain an abstract class?**

**Answer.** An abstract class is a template definition of methods and variables of a class (category of objects) that contains one or more abstracted methods. Abstract classes are used in all object-oriented programming (OOP) languages, including Java

**27.Explain operator overloading?**

**Answer.** Operator overloading is a technique by which operators used in a programming language are implemented in user-defined types with customized logic that is based on the types of arguments passed.  
  
Operator overloading facilitates the specification of user-defined implementation for operations wherein one or both operands are of user-defined class or structure type. This helps user-defined types to behave much like the fundamental primitive data types. Operator overloading is helpful in cases where the operators used for certain types provide semantics related to the domain context and syntactic support as found in the programming language. It is used for syntactical convenience, readability and maintainability.

**28.Define different types of arguments? (Call by value/Call by reference)**

**Answer.** In the case of Call by Value, when we pass the value of the parameter during the calling of the function, it copies them to the function's actual local argument. In the case of Call by Reference, when we pass the parameter's location reference/address, it copies and assigns them to the function's local argument.

**29.Explain the super keyword?**

**Answer.** The super keyword refers to superclass (parent) objects. It is used to call superclass methods, and to access the superclass constructor. The most common use of the super keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

**30. Explain method overriding?**

**Answer.** Method overriding, in object-oriented programming, is a language feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its superclasses or parent classes.

**31. Difference among overloading and overriding?**

**Answer:**

| **Overriding** | **Overloading** |
| --- | --- |
| Implements “runtime polymorphism” | Implements “compile time polymorphism” |
| The method call is determined at runtime based on the object type | The method call is determined at compile time |
| Occurs between superclass and subclass | Occurs between the methods in the same class |
| Have the same signature (name and method arguments) | Have the same name, but the parameters are different |
| On error, the effect will be visible at runtime | On error, it can be caught at compile time |

**32.Whether static method can use non-static members?**

**Answer**: A static method can only access static data members and static methods of another class or same class but cannot access non-static methods and variables. Also, a static method can rewrite the values of any static data member.

**33.Explain a base class, sub class, super class?**

**Answer:** Sub Class/Child Class: Subclass is a class which inherits the other class. It is also called a derived class, extended class, or child class. Super Class/Parent Class: **Superclass is the class from where a subclass inherits the features**. It is also called a base class or a parent class.

**34. Write in brief linking of base class, sub class and base object, sub object.**

**Answer.** In Java, all non-static methods are based on the runtime type of the underlying object rather than the type of the reference that points to that object. Therefore, it doesn’t matter which type you use in the declaration of the object, the behavior will be the same.

There are two approaches to refer a subclass object. Both have some advantages/disadvantages over the other. The declaration affect is seen on methods that are visible at compile-time.

1.First approach (Referencing using Superclass reference): A reference variable of a superclass can be used to a refer any subclass object derived from that superclass. If the methods are present in SuperClass, but overridden by SubClass, it will be the overridden method that will be executed.

2.Second approach (Referencing using subclass reference) : A subclass reference can be used to refer its object.

**35. Explain an interface?**

**Answer.** An interface in Java is a blueprint of a class. It has static constants and abstract methods. The interface in Java is a mechanism to achieve abstraction

**36. Explain exception handling?**

**Answer.**  Exception handling is the process of responding to unwanted or unexpected events when a computer program runs. Exception handling deals with these events to avoid the program or system crashing, and without this process, exceptions would disrupt the normal operation of a program.

**37 Explain the difference among structure and a class?**

**Answer:** Basically, a class combines the fields and methods(member function which defines actions) into a single unit. A structure is a collection of variables of different data types under a single unit. It is almost similar to a class because both are user-defined data types and both hold a bunch of different data types

**38. Explain the default access modifier in a class?**

**Answer**. Default access modifier means we do not explicitly declare an access modifier for a class, field, method, etc. A variable or method declared without any access control modifier is available to any other class in the same package.

**39. Explain a pure virtual function?**

**Answer.** A pure virtual function or pure virtual method is **a virtual function that is required to be implemented by a derived class if the derived class is not abstract**. Classes containing pure virtual methods are termed "abstract" and they cannot be instantiated directly.

**40. Explain dynamic or run time polymorphism?**

**Answer.** Dynamic polymorphism is a process or mechanism in which a call to an overridden method is to resolve at runtime rather than compile-time. It is also known as runtime polymorphism or dynamic method dispatch. We can achieve dynamic polymorphism by using the method overriding.