BSIT (II) OOP [redacted] Prof Sami

1.Object:-

An object has characteristics or attributes; an object has actions or behavior, Specially, an object is an entity that consists of **Data(the attributes)**, and **methods** that use or manipulate the data(the behavior).

2.Encapsulation:-

Encapsulation is defined as the language feature that packages attributes and behavior into a single unit that is, data and methods comprise a single entity.

3.Class:-

A class is a template or blueprint, from which objects are created.

4.Concatenation:-

Concatenation is the process of joining, connecting, or linking strings together.

5.Q: Can a program call a default constructor?

Ans: A program cannot call the default constructor directly; it is invoked via the new operator.

Compose By: Muhammad Bilal

6.Constructor:-

- A constructor is automatically executed each time an object of the class is instantiated.
- It is used for initialize the value.

7.Static Data or Class Variable:-

A static variable belongs to the class and not to any particular object ; a class or static variable is shared by all objects of the class.

8.Use of This Keyword:-

- Using the keyword this, one constructor calls another constructor.
- If one constructor calls another constructor, on other statements can precede that call.

9.Inheritance:-

Inheritance is the mechanism that allows us to reuse the attributes and methods of one class in the implementation of another class.

10.Over riding:-

A type of polymorphism defined as same name and same arguments.

11.Access specifier:-

Access specifier is used to shows visibility level.

Access specifier

Private

A private variable or method is visible only to its defining class.

Public

A public variable method is visible to any class.

Protected

A protected variable or methods is visible to its defining class and all its subclasses, as well as any other classes in the same package.

12.Subclass:-

A subclass inherits all public and protected methods of a base class unless the subclass overrides a methods, thus providing its own implementation.

- A subclass does not inherit the constructor of base class.

13.Abstract class:-

An abstract class is a class that cannot be instantiated. However, an abstract class can be inherited.

14.Properties of Abstract class:-

Compose By: Muhammad Bilal

- An abstract class, may contain abstract method.
- A class having a single or many abstract methods would be an abstract class.
- All abstract classes and methods are public.
- To be of any use, an abstract class must be extended.

15.Upcasting:-

Upcasting is a language feature that allows a base type reference to refer to an object of a derived type.

16.Downcasting:-

Downcasting means casting an object to a derived or more specialized type.

- To invoke a derived class methods using a base class reference, a downcasting is necessary.

17.Interfaces:-

An interfaces can be used to achieve polymorphism.
An interface is a named collection of static constants and abstract methods. An interface specifies certain actions or behaviours of a class but not their implementations.

- All methods of an interface are public.
- All methods of an interface are abstract, that is, there are no implementations at all.
- An interface has no instance variable.

- A class that implements an interface must implement all the methods of the interface, or be tagged as abstract.
- A class may extend one class as well as implement any number of interfaces.

18.Upcasting to an Interface:-

A derived class can be upcasting to any one of its in terfaces.

19.Polymorphism:-

In contrast to inheritance, polymorphism underscores the differences of class behaviour in an inheritance hierarchy.

- Polymorphism allows you to extends yours classes with ease.
- Polymorphism derived from Greek words

Polus → many shapes.

morphe → many forms.

20.Over Loading:-

A type of polymorphism defined as same name but different arguments.

21.Methods OverLoading:-

Mehtods overloading, a form of polymorphism, is also known as ad-hoc polymorphism.

22.Dynamic (or late) Binding:-

A third form of polymorphism, dynamic(or late) binding, accentuates the behavioral differences among objects of different classes in a hierarchy.

Polymorphism via dynamic or late binding refers to choosing the appropriate method not at compile time, but at runtime.

23.Q: How dynamic binding works?

Ans: The real type or actual type of a reference variable is the type of the object that is created by the new operation.

24.Wrapper classes:-

- Java wrapper classes provide genuine classes for each primitive data type.
- All wrapper classes, except Boolean, implement the comparable interface.
- As you know, a variable can be either a reference or
- A primitive (double, float, int, char, boolean, etc).

25.Propertities of Wrapper classes:-

- Each numeric class (Integer, Long, Short, Byte, Double, and Float) has one argument constructor that accepts an arguments of the corresponding primitive type.
- Each wrapper class, except character, has a second constructor that accepts a string argument.

26.Exceptions:-

An abnormal condition that occurs at runtime is called an exception.

27.Exception Handling:-

- Java exception class and its subclasses provide an automatic and clean mechanism for handling exceptions.
- To handle exceptions uniformly and efficiently, Java provides the try-throw-catch construction.

28.When an exception occurs:-

Generally speaking, when an exception occurs,

- An exception object that holds information about the exception is instantiated.
- The exception object is passed, or thrown, to a section of code called a catch block that handles the exception.

29.Multile Catch Blocks:-

Multiple catch blocks should be written in order from most specific to least specific exception.

30.Unchecked exception:-

- An unchecked exception, such as an out of bounds array index, is one that usually cannot be handled during runtime.

Compose By: Muhammad Bilal

- Catching an unchecked exception is the programmer's choice.

31.Checked Exception:-

An exception that is not unchecked is called a check exception. A checked exception is one from which a method can reasonably be excepted to recover.

32.Finally Block:-

A finally block is a block of code that always executes, regardless of whether or not an exception is thrown. A finally block is paired with either a try-catch pair or a try block.

33.Container:-

A container is an object that holds components.

34.Component:-

A component is an object that can be displayed on screen.

- Every component, such as a button a checkbox, a textbox, or a window is an object belonging to some class that extends component, these components are placed in containers.

35.JFrame:-

A JFrame encapsulates what you normally think of as a "window" and it is the primary container used in all our applications.

36.Centering a Frame:-

To center a frame on a screen of any size, use method of Java Toolkit and the Dimension classes.

37.Layout Managers:-

A layout manager is an object that arranges components in a container such as a frame. The layout manager classes implement the Layout Manager interface.

38.BorderLayout:-

If no region is specified, a BorderLayout layout manager places a component in the center region. Only one component can be placed in a region, and components are resized to fit the region.

39.FlowLayout:-

An object belonging to Flowlayout arranges components horizontally in a container, left to right, row to row, in the order in which they are added to the container.

40.GridLayout:-

The Gridlayout layout manager arranges the components of a frame in a grid of specified dimensions, left to right, top to bottom, row by row.

41.Panels:-

A panel is an invisible container used for arranging and organizing components.

Compose By: Muhammad Bilal

BSIT

OOP

Prof Sami

42.Label:-

A Label object is a component for placing text in a container. A label displays a single line of read-only text. The text can be changed by the application, but a user cannot edit it directly.

43.The paint () & paint Component() methods:-

The paint(...) and for paint Component(...) methods are invoked automatically whenever the system determines that a component should be drawn or redrawn on the screen.

44.Graphics Context:-

A graphics context represents a drawing destination. It contains drawing parameters and all device-specific information that the drawing system needs to perform any subsequent drawing commands.

Compose By: Muhammad Bilal