Page No.
Date

Constructor.

1) Constructor is used to initialized object.

a) Parameterized Con.
b) Non - parameterized Constructor.
c) Default Constructor.

Properties

1) Constructor have same name as class nam
2) Does not any value.
3) Constructor call only one time for one object.

Polymorphism

1) Polymorphism is considered one of the Imp. features of OOP's.
2) It allow us to perform a single action in different ways.

Ex. water is a liquid at normal temperature but it can be changed to solid when it frozen. And water changes to gas when it is heated at its boiling point.

∴ Same water ~~exhibit~~ exhibiting different role in polymorphism.

Types

1) Compile Time Polymorphism
   - Achieved by function Overloading

2) Runtime Polymorphism
   - Achieved by function Overriding

Inheritance

* Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object.

1) Single Inheritance

Base Class
↑
Derived Class

when a class inherits another class it is called single inheritance.

Ex. single Inheritance

```
class Animal {
    void eat() {
        system.out.println("eating");
    }
}

class Dog extends Animal {
    void bark() {
        system.out.println("barking...");
    }
}

class TestInheritance {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.bark();
        d.eat();
    }
}
```

output : barking...
eating...

2] Multilevel Inheritance

It is chain of inheritance means a ~~der~~ parent class is derived from another class. is called Multilevel Inheritance

Ex.

```
Class Animal {
        void eat() {
                System.out.println("eating ...");
        }
}

class Dog extends Animal {
        void bark() {
                System.out.println("barking ...");
        }
}

class BabyDog extends Animal {
        void weep() {
                System.out.println("weeping ...");
        }
}

class TestInheritance2 {
        public static void main(String args[]) {
                BabyDog d = new BabyDog();
                d.weep();
                d.bark();
                d.eat();
        }
}
```

/outputs eating ...
barking ...
weeping ...

Page No.
Date

3] Hierarchical Inheritance

when two OR more classes inherites a single class is called Hierarchical

Ex.

```
class Animal {
    void eat() {
        system.out.println("eating ---");
    }
}

class Dog extends Animal {
    void bark() {
        system.out.println("barking ---");
    }
}

class Cat extends Animal {
    void meow() {
        system.out.println("meowing
    }
}

class TestInheritance3 {
    public static void main(String[] args)
        Cat c = new Cat();
        c.meow();
        c.eat();
        Dog d = new Dog();
        d.bark();
        d.eat();
    }
}
```

output :

Page No.
Date

4] ~~why~~ Hybrid Inheritance

5] why multiple inheritance is not supported in java.?

- To reduce the complicity
- To avoid ambiguity
- to make it simple.

Date

Encapsulation

Prequisite : Packages and Access Modifier

Packages : Java Packages is a group of similar types of classes, interfaces and sub-packages

Types.
1) Build In Packages
2) User Defined Packages

Types of Access Modifiers in Java.

1) Default : No keyword Required

2) Private :
- The methods or data members declared as private are accessible only ~~whi~~ within the class in which they are declared.
- Any other class of the same package will not be able to access these member
- Private means "Only visible within the enclosing class."

Page No.
Date

* Top level classes OR Interfaces can not be declared as private. because,
  - private means "only visible within the enclosing class"
  - Protected means "only visible within the enclosing class and subclasses."

3] Protected

The methods or data members declared as protected are accessible within the same package OR subclasses in different packages.

4] Public

* Encapsulation provides Data Hidding

Page No.
Date

Encapsulation: It is a process of wrapping code and data together into single unit.

We can create a fully encapsulated class in java by making all the data members of the class private.

By providing only a setter OR getter method we can make the class read-only OR write only.

* Getter and Setter make the programmer convienient in setting and getting the value for a perticular value.

Getter: Returns the Value.

Setter: Setter sets OR updates the value

Abstraction

Abstraction in Java is the process in which we only show essential details or functionality to the user.

The non-essential implementation details are not displayed to the users.

* Abstraction is achived by interfaces and abstract classes.
we can achieve 100% abstraction using interfaces.

Properties of Abstract Classes and Abstract Methods

1) Abstract class is class that is declared with abstract keyword.
2) Abstract class contain both abstract and non abstract method.
3) ** No object of Abstract class.
Abstract class can not be directly instantiated with the new keywords.
4) It can have constructors and static methods

Interfaces In Java.

1) It is a mechanism to achieve Abstraction
2) Interfaces can have abstract methods and variables.
3) Using interfaces we can achieve multiple inheritance.
4)

Class	Interfaces.
1) Create object and allow to instantiate variables	Can't Instantiate and Create object.
2) Contain implemented (body) method	Can not contain implemented method
3) Public, static, private and default Access specifier are there	Public.

Exception Handling in Java.

Page No.
Date

1) The Exception Handling in Java is one of powerful mechanism to handle the runtime errors so that normal flow of the application can be maintained.

Exception is an event that disrupts the normal flow of program.

Hierarchy of Java Exception Classes.

The java.lang.Throwable class is the root class of Java Exception hierarchy inherited by two subclasses. Exception and Error.

The hierarchy of Java Exception classes is given below.

There types of Errors which are follows.

1] Checked Error.

The classes that directly inherite the throwable class except Runtime Exception and
This errors are known as checked exceptions.

Ex. IO Exception
SQL Exception

checked exceptions are checked at compile time.

2] Unchecked Exception

The classes that inherit the Runtime Exception are known as unked exceptions.

Ex. Arithmetic Exception
NullPointer Exception
ArrayIndex OutOF Bound Exception

Unchecked exceptions are not checked at compile-time but they are ~~the~~ checked at runtime.

3] Error

Error is irrecoverable.

Ex Out Of Memory Error.
Virtual Machine Error
Assertion Error

Keywords	Description
1) try	The "try" keyword is used to specify a block where we should place an exception code. try block must be followed by either catch OR ~~fitt~~ finally. we can't use try block alone.
2) Catch	The "catch" block is used to handle the exception.
3) finally	The "finally" block is used to execute the necessary code of the program. It is executed whether an exception is handled or not.
4) throw	The "throw" keyword is used to throw an exception.
5) throws	throws is used to declare exceptions. It specifies that there may occur an exception in the method. It doesn't throw an exception.

Multithreading

Multithreading in Java is a process of executing multiple threads simultaneously.

- Thread is lightweight sub-process, The smallest unit of processing.