Introduction to Java

- Java is a programming language and a platform. Java is a high level, robust, object-oriented and secure programming language.

- Java was developed by Sun microsystem (which is now the subsidiary of Oracle) in the year 1995. James Gosling is known as the father of Java. Before Java, its name was Oak. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.

JDK : Java Development Kit

- The Java Development Kit (JDK) is a Software development environment which is used to develop java applications and applets. it physically exists. it contain JRE + development tools.

- There are 3 type of platforms
  1) Standard Edition Java platform
  2) Enterprise Edition Java platform
  3) Micro Edition Java platform.

* JVM :- Java virtual machine

- Jvm (Java virtual machine) is an abstract machine. it is a specification that provides runtime environment in which Java bytecode ~~can~~ can be executed

- what is Jvm
  1. A Specification where working of Java Virtual machine is specified. But implementation provider is indepeded to choose the algorithm. its implementation has been provided by oracle and other compaines.

* JRE :- Java Run-time Environment

- Java Run-time Environment (JRE) is the part of the Java Development Kit (JDK). It is a freely available software distribution which has Java class library, specific tools, and a standalone Jvm. It is the most common Environment. avaiable on devices to run ~~time~~ Java program's. The Source Java code gets compiled and converted to Java bytecode. If you ~~watch~~ wish to run this bytecode on any platform, you require JRE. The JRE loads ~~as a layer on the top of the operating system.~~ classes, verify acess to memory, and retrieves the System resources. ~~JRE~~ JRE acts as a layer on the top

of the operating system

* Data types

Data types specify the different sizes and values that can be stored in the variable

There are two types of data types

1) Primitive data types:-

The primitive data types include boolean, char, byte, short, int, long, float and double

2) non - primitive data types:- The non-primitive data types includes classes, interfaces and Arrays.

* variables:-

i) A variable is a contain which holds the value while the Java program is executed. A variable is assigned with a data type.

ii) variable is a name of memory location. There are three types of in java

1) local.
2) instance
3) static

1) local :-

A variable Declared inside the body of the method is called local variable.

2) instance :-

A variable declared inside the class but outside the body of the method, is called an instance variable. it is not Declared as static. It is called an instance variable because its value is instance-specific and is not shared among instances

3) static :-

A variable that is declared as static variable. it cannot be local - you can create a single copy of the static variable and share it among all the instances of the class. memory allocation for static variables happens only once when the class is loaded in the memory.

Scanner class

this is inbuit class which is available in java.util.scanner package, we use this class to take user input from command line interface. there are multiple methods available to take user input according to a variable datatype. such as nextInt(), nextline();

syntax:

```
Scanner sc = new Scanner (System.in);

sc.nextInt();
```

* Classes & object

* Class :-

in Java, class is a group of similar object. it is a template from which object are created. it can have fields, methods, constructors etc.

Syntax or ~~ex:~~

```
class <class_name> {
    field;
    method;
}
```

* object :-

- In Java, object is a real world entity

for Ex:- chair, car, pen, mobile, laptop

- In other words, object is a entity that has state and behaviour means functionality.

- object is a runtime entity, it is created at runtime

- object is an instance of class. All the members of the class can be accessed through object.

Syntax:- Student S1; // creating an object of Stude

:- Access specifiers :-

Access modifiers :- The acess modifiers in java specifies the accessibility or scope of a field, method, constructor or class. we can change the access level of fields, constructor methods and class by applying the acess modifiers on it.

1) private :- The access level of a private modifier is only within the class. it cannot be accessed from outside the class.

2) Default :- The acess level of a default modifier is only within the package. it cannot be accessed from outside the package. if you do not specify any access level it will be the default.

3) protected :- the access level of a protected modifier is withing the package and outside the package through child class. if you do not make the child class, it cannot be accessed from outside the packages.

4) public :- The access level of a public modifier is everywhere. it can be accessed from within the class, outside the class, within the package and outside the package.

* Constructor :

It is a special type of function used to create objects. When we create object then the constructor is invoked implicitly. There are 2 type's of constructor. default constructor & parameterized constructor. when we want to pass data into the ~~instru~~ instance at the time of creation then we use constructor.

ex:- class Demo() {
public:
Demo() {
}
}

* Default constructor
A constructor which has no argumen in known ~~as~~ default constructor. it is invoked at the time of creatir objects.
ex:-

Public class Defcon {
class Def {
public:
Def() {

```
System.out.println("Default constructor");
    }
}
public static void main(String args[])
{
    Def d1;
}
```

* Parameterized constructor :-

A constructor which has parameter is called parameterized constructor it is used to provide different value to distinct object.

ex.-

```
public class paracon{
    class para {
        public :
            para() { }

            para(int a, in b) {

System.out.println(a+b);
    }
}
public static void main(String args[])
{
    para p1;
    p1 = para(3,5);

}
```

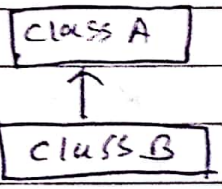
Inheritance :-

In Java inheritance is a process in which an object acquires all the properties and behaviours of its parent automatically in such way, you can reuse, extend or modify the attributes and behaviours which are defined in other class.

- Advantages
  code reusability

- Types of Inheritance
  1) Single inheritance
  2) multilevel inheritance
  3) Hierarchical inheritance

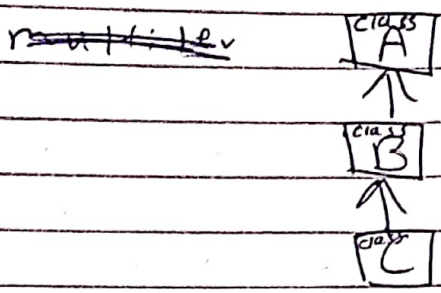
1) Single Inheritance.



Single inheritance is process of deriving a new class that inherits the attributes from two class or more classes

ex:-
```
class A {
    public :
};
class B extends A {
    public :
};
```

2) multi level inheritance :-

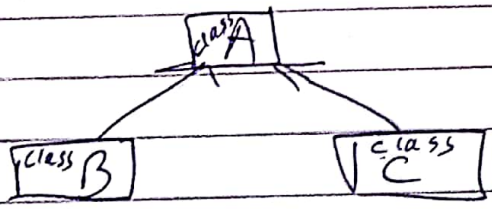


multiple inheritance is a process of deriving a class from another derived class

ex:-

```
class A {
    public:
};
class B extends A {
    public:
};
class C extends B {
    public:
};
```

3) Hierarchical inheritance :-



Hierarchical inheritance is defined as the process of deriving more than one

class from a bass class.

ex:-

```
Class A {
    public:
};
Class B extends A {
    public:
};
Class C exteds A {
    public:
};
```

* Polymorphism:-

polymorphism allows object of Different classes to be treated as object of a comman parent class it's achieved through virtual function and can be implemented using inheritance and function overloading.

* function overloading :-

in ~~C++~~ Java allows you to nave multiple function with the same nume but Different purumeters or parametes types enabling you to perform different actions based on the input received.

* Function overriding :-

in Java occurs when ~~implementation~~ a derived class provides a specific implementation for a method that is already defined in its base class. It involves redefining a function in the Derived class. The same signature (name & parameters) as the one in the base class to customize its behaviour.

* Encapsulation :-

In Java, Encapsulation bundles data and methods within a class, limiting direct access to data and enabling controlled access through public method for enhanced security and ~~metho~~ modularity

getter ()

Setter ()

* Abstraction :-

Abstraction in Java involves hiding implementation details & exposing only essential features. it is achieved through ~~abstract~~ ~~ASI~~ ~~Abstract~~ Abstract classes and interfaces, allowing the creation of generic structures. By promoting code flexibility, abstraction enhances software maintainability and extensibility.

* Abstract class :-

In Java, an abstract class serves as a blueprint for other class and cannot be instantiated on its own. it may contains both abstract (unimplemented and concrete (implemented) methods, Sub class must provide implementations for abstract methods, promoting code consistency and structure.

* interface :-

In Java, an interface defines a contract of abstract methods without providing implementations. Classes implement interfaces to adhere to the specified behaviour. interfaces support multiple inheritances, enabling a class to implement multiple interfaces.

* Static Keywords :-

in Java is used for memory management mainly. We can apply Static keyword with variables, methods, blocks & nested classes.

1) Static variable
2) static method
3) Static block
4) nested classes

1) static variables :-

in Java, a static variable is a class variable that belongs to the class rather than to instances of the class. it's shared amoung all instances of the class. You declare a static variable using the 'static' keyword.

2) Static method :-

in Java, a ~~Stic~~ Static method belongs to the class rather than an instances of the class. You can ~~call~~ call a static method on the class itself, without creating an instance of that class. TO declare method, use the 'static' keyword in the method signature.

3) Static block :-

in Java, a static block is used for static initialize of class and is executed only once when the class is loaded

4) Varargs () :-

in Java is a method that takes a variable number of arguments. variable Arguments in Java simplifies the creation of of methods that need to take a variable number of arguments

* Math class :-

in Java provides static methods for common mathematical operations, ~~such as~~ it includes functions like min(), max(), avg, sin(), cos(), tan(), round(), ceil(), floor(), abs() etc. for finding the maximum of two values. Developers can use these methods without creating an instance of the math class.

* String class :-

The String class is a data type in programming used to represent and manipulate text. it typically includes methods for concatenation, substring extraction, and lenght retrieval. Strings are immutable in some languages, meaning their values cannot be changed after creation.

imp Multithreading :-

in Java is a process of executing multiple treads simultaneously.

A thread is a light weight. sub-process the smallest unit of processing. multiprocessing and multithreading, both are used to achieve multitasking.

~~1) wait~~

Java thread Methods

1) wait()
2) sleep()
3) Notify()
4) resume()
5) Suspend()
6) Stop()

1) wait() :-

In Java is used to pause the execution of a tread until another thread signals that it can ~~seso~~ resume.

2) sleep() :-

This is a method used to sleep the thread for some milliseconds time.

3) Notify() :-

This method of Thread Class is used to wake to wake up a single thread. This method gives the notification for only one Thread which is waiting for a particular object.

4) resume() :-

This is a method used to resume the suspended Thread.

5) Suspend() :-

This method is used to stop the thread execution and start it again when a certain event occurs. This method allows a thread to temporarily ceasve execution. The suspended Thread can be resumed using the resume() method.

6) Stop() :-

This method of thread class terminates the thread execution. once a thread is stopped, it cannot be restarted by Start() method.

* Garbage Collector :-

in Java automatically manages memory by reclaming memory occupied by objects that are no longer referenced, freeing resources for reuse. it runs in the background, identifying and removing unreferenced objects, preventing memory leaks and allowing efficient memory utillization.

1) finalize () method
2) gc () method

1) finalize():-

This method is invoked each time before the object is garbage collected. This method can be used to perform cleanup processing. This method is defined in object class as:

2) gc ():-

This method is used to invoke the garbage collector to performe cleanup processing. The gc() is found in System and Runtime classes.