Java introduction

Java is a simple and most widely used programming language.

why?

Free ware and opensource.

It is platform Independent that is program written in one operating system is capable of running in all other.

operating System due to byte code concept

It runs multiple application at a time.

Main feature and Advantage of Java.

1.) platform independent.

During the compilation the java program is converted into byte code, Bytecode can be runned in JVM of any platform. so code developed in one platform capable running in all other platform.

2.) Open Source:-

A program in which source code is available to general public for use and/or modification from its original design at free of cost is called open source.

3. Multithreading

Java supports multithreading. It enables a program to perform several task simultaneously.

4. More Secure

It provides virtual firewall between the application and the computer. So it doesn't grant unathorized access.

5. portable → write once, runs anywhere.

JDK → Java Development kit

if we want to create any application in Java JDK have to be installed in our system.

JVM → Java virtual machine

It is mainly used to allocate the memory

JRE → Java Runtime Environment

It is a predefined class files(or) library files.

oops Concept

object oriented programming structure.

it is a method of implementation in which program is organised as collection class, methods, & objects.

principles:

- class
- method
- object
- inheritance
- polymorphism
- abstraction
- encapsulation.

class:

class is the collection of objects and methods.

method:

set of action to be performed // business logices.

object:

it is a runtime memory allocation

it is used to call the methods.

syntax

```
className objectName = new className();
```

Standard Notation:

1) pascal Notation

Each word first letter, must be in capital letter.

project name, class name.

2. Camel Notation

First word first letter should be in small letter, remaining word first letter start with capital letter,

method name, object name, variable name.

project
package
class
method
object.

Data Types

It specifies the type and size of variable.

variable is used to store the values.

Syntax:

```
Datatype variableName = value;
```

Datatype	Size(Byte)		Wrapper class.
byte	1	2	Byte
short	2	4	Short
int	4	9	Integer
long	8	19	Long.

→ Used for store whole number 0 ... n

float	4	Float
double	8	Double

→ Used for store decimal number and fractional number.

char('A')

String("alphabets, numerals, special characters")

Character

boolean(true/false)

Boolean

Range calculating formula

1 byte = 8 bits

Range = $-(2^{n-1})$ to $(2^{n-1})-1$

n ---> bits

byte = $-(2^7)$ to $(2^7)-1$

= -128 to 127

Wrappor class

classes of datatype is called wraprerclass

It is used to convert datatype into object

In data type we have two types

1.) primitive datatype :--> byte, short, int, long, float, doub
char, boolean

2.) non-primitive datatype ---> String

Inheritance

we can access one class property from another class by using extends keyword.

Adoantage

* Reusable code purpose
* Memory waste is low

Single Inheritance.

Combination of one parent class and one child class

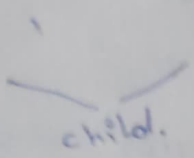
parent

↓

child

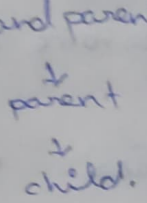
Multiple Inheritance.

more than one parent class accessing the child class parallely at a time.



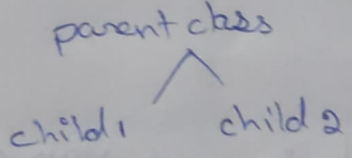
Multilevel Inheritance.

more than one parent class accessing the child class in tree level structure.



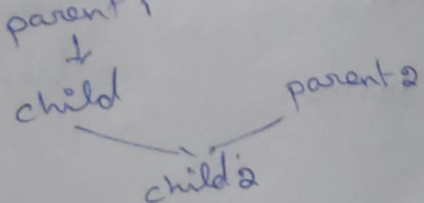
Hierarichal Inheritance.

Combination of one parent class and more than one child class.



Hybrid Inheritance.

Combination of single and multiple Inheritance.



multiple inheritance

Does not work in java with use of extends keyword due to

1.) priority problem

2.) Syntax error

3.) compile time error

Achive the multiple inheritance

by using Interface we can achieve the inhe

Syntax:

access specifier class childclass name extends parentclass

- access specifier — private, public
- childclass name — sub class (subclass)
- extends — keywords
- parentclass — super class (superclass)

parent class --- where we are GETTING the property

child class --- where we are ACCESSING the property

polymorphism

Executing method in more than one form or as completing the same task in many ways.

poly --> many

morphism --> forms.

Method overloading (compile time polymorphism / static birding / static polymorphism).

Same class
same method
different arguments

Arguments deperds on datatype.
Arguments deperds on datatype count
Arguments deperds on datatype order.

Method overriding (runtime polymorphism / dyramic birding / dynamic polymorphism)

Same method
same arguments
different class

Abstraction

Hiding the implemantation details or method business logic details

Types of abstraction:

*) partial abstraction
*) fully abstraction.

Partial abstraction (abstract class):

Contains both abstract and non abstract methods

We can create object

Contains keywords extends

Fully abstraction (Interface):

Contain only the abstract methods

We can create object

Contains keywords implements.

abstract ---> the method should not contain any business logic.

non abstract ---> the method which contain business lo

Conditional Statement

if

```
if(condition) {
    // statement
}
```

Condition true, it will executed otherwise it won't

if else

```
if(condition){
    //true block statement
}else{
    //false block statement
}
```

whether the condition is true or false it will executed

more than one condition

And - BITWISE &, LOGICAL AND &&

con 1	con 2	result
T	T	T
T	F	F
F	T	F
F	F	F

OR - Bitwise |, Logical OR ||

con 1	con 2	result
T	T	T
T	F	T
F	T	T
F	F	F

Laden If / nested if else

```
if (condition) {

} else if (condition) {
} else if (condition) {
    :
} else {
```

Control statement

control the flow of execution

1) conditional statement
2) Looping statement
3) jumping statement