

### Java installation procedure:

All program/ press window button -> Enter cmd

**Step1** : Command to check java version: **java -version**

**Step2** : check operating system: right click on This PC/My Computer--> properties--> Device Specifications--> system type

OS 32 bit--> **8** (1.8)

OS 64 bit--> **11** - 18

**Step3** : how to download java

search download java 1.8-->click on oracle.com--> scroll down up to java 8/11--> windows --> click on 32/64 related file to download java

**Step4** : install downloaded java file by double clicking on file, click on next – next until process completed, don't change any configuration while installing

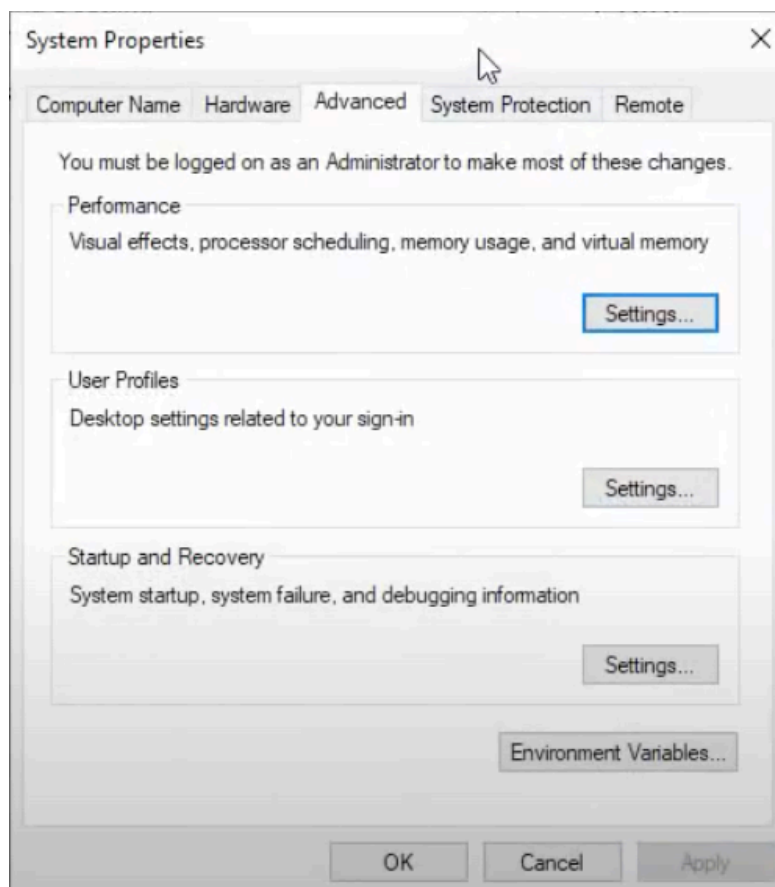
**Step5** : set java path:

**Step 5A** : Copy java path

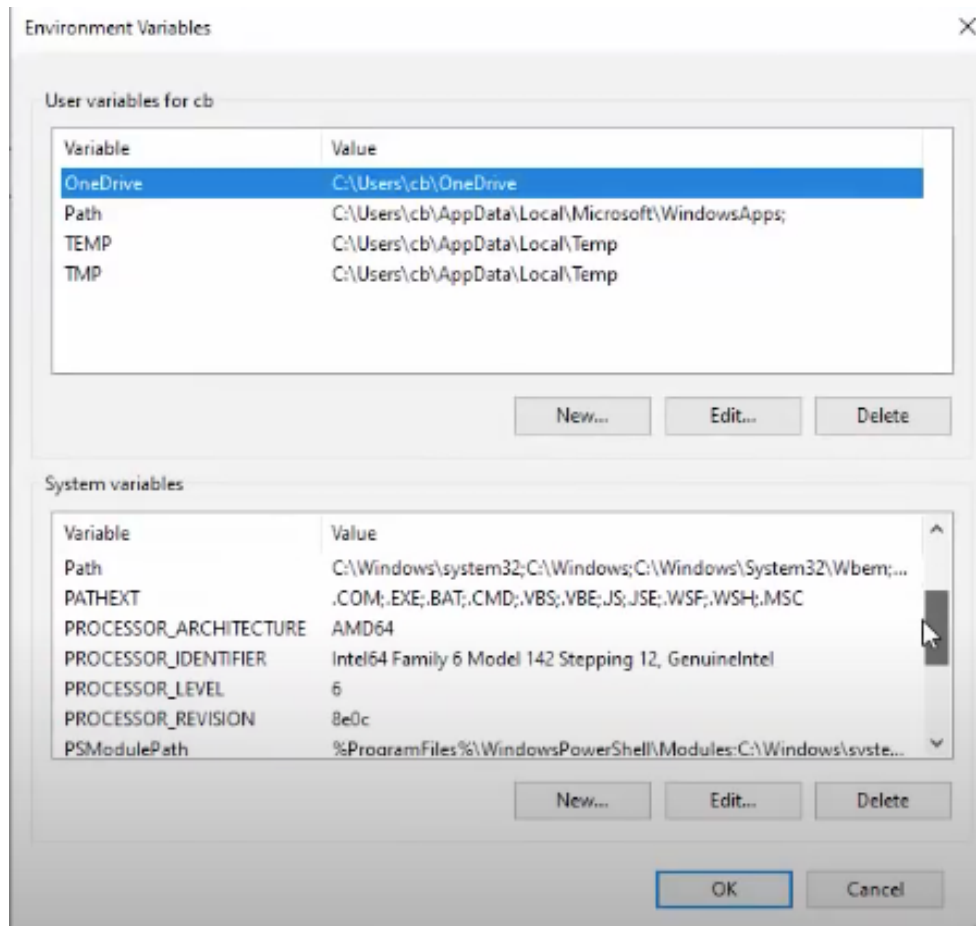
My computer --> open C drive-->program files-->java-->JDK/JRE( JDK preferred)-->bin folder-->copy bin folder path (control + C)

**Step 5B** : Set java path

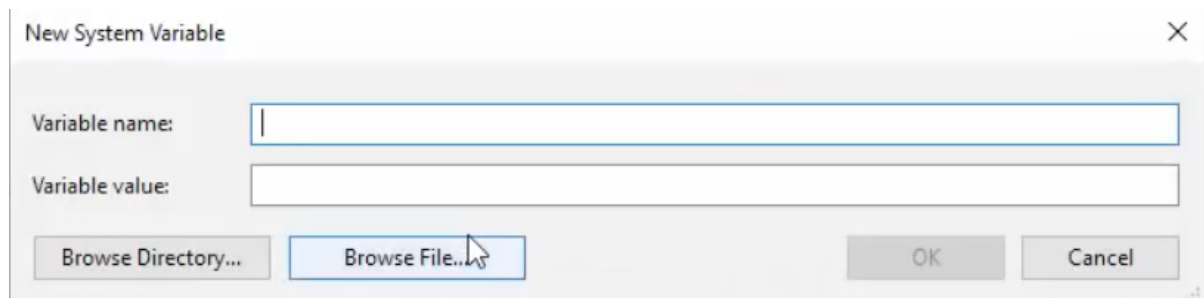
All program/ press window button ---> Search **edit environment variable**



Click on **environment variable** button



In System variable click on **new** button

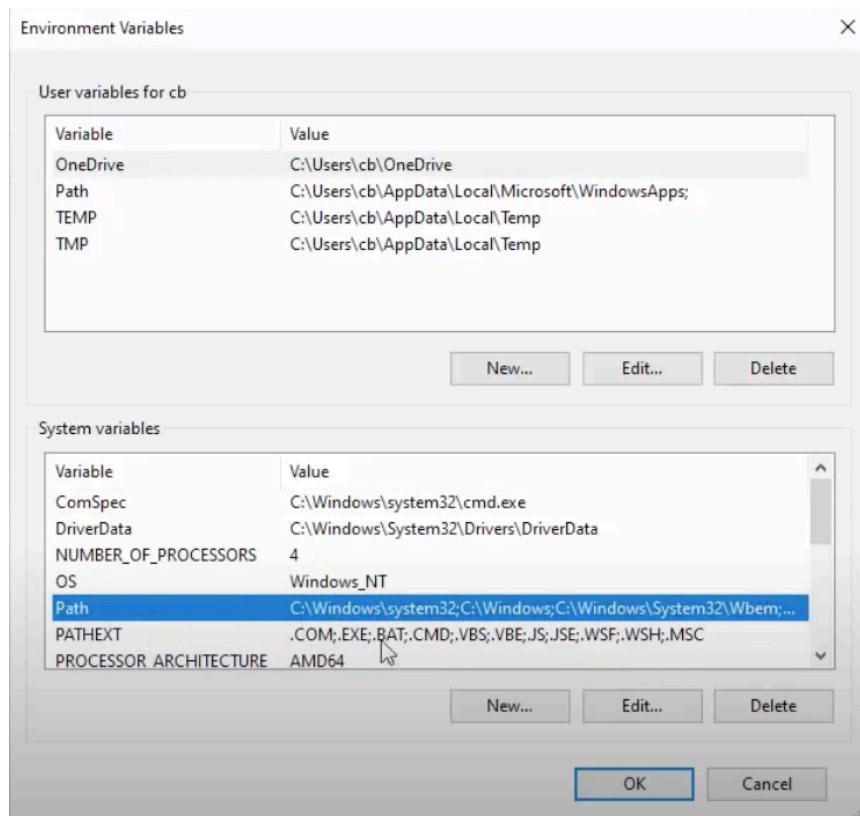


Variable name = JAVA\_HOME

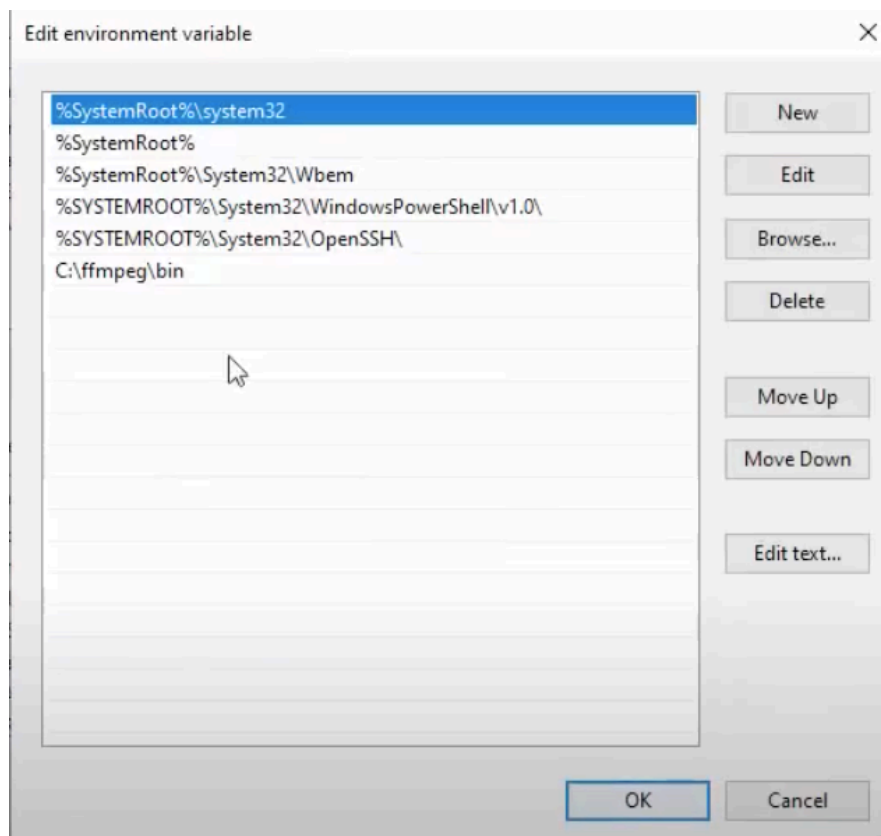
Variable value = c:\Program files\Java\jdk1.8.0\bin

Press OK

Select path variable in System variables



Click on **edit** button



Click on new  
Enter %JAVA\_HOME%

Press OK  
Press OK

**Case1** : Already path variable exist --> click on path variable-->edit-->new—Enter>  
%JAVA\_HOME% --> ok-- ok

**Case2** : No path variable--> click on new --> enter variable name i.e. -"path"--> variable  
value - %JAVA\_HOME% --> ok-- ok

**Step6** : Verify java installed or not ?  
check for java version  
execute command --> java -version

**Error message** : version 1.80 \_261 of the JVM is not suitable for this product version 11  
or greater is required --> install java 11 or install eclipse old version

**Check PC/laptop configuration** : Right click on This PC/ My Computer --> Properties--  
> System type

### **Eclipse installation procedure:**

**Step1** : Search download eclipse -->  
click on <https://www.eclipse.org/>-->  
click on Download packages  
Click on releases  
Select your version (2018-09)  
Click on R packages  
Click on IDE for java developer specific version

**Step2** : Open downloads folder --> unzip eclipse file ---> open eclipse folder--> double  
click on eclipse application file (blue colour icon)-->  
it should ask for workspace path--> keep as it is(default path)-->select checkbox--  
>launch--> welcome page

Compatibility issue for windows 7 32 bit OS:  
the version of this file is not compatible with the version of windows you are running  
check your computer info---.  
java 8 with eclipse version helious

## -----Core Java-----

### **Section-1 (basic java)**

1. Variable & data types
2. Keywords & Identifiers
3. Methods
3. Types of variable
4. Constructor
5. Control statements
6. Loops

### **Section-2 (OOPs)**

1. Inheritance
2. This & super keyword
3. Access specifiers
2. Polymorphism
4. Abstract class & concrete class
5. Interface & Implementation class
6. generalization
7. casting
8. encapsulation --> selenium framework
9. Abstraction

### **Section-3**

1. String Class --vvp
2. Array
3. Exception handling
4. Collection
5. Map
5. Pattern programs
6. Logical programs

## Java:

James Gosling developed the Java platform at Sun Microsystems at 1991 and the Oracle Corporation later acquired it.

Java is general purpose, concurrent, object oriented, class based and runtime environment (JRE) which consist of JVM which is cornerstone (base) of java platform

- It is object oriented language with advance and simplified features
- It is free to access and can run on all platform
- It is concurrent where we can execute many statement instead of sequential execution
- It is independent programming language which follow the logic of "Write once, Run anywhere" i.e. complied code can run on all platform which support java
- All code converted in bytecode after compilation which is unreadable for human
- It enable to develop virus free and tamper free application
- It is purely object oriented, platform independent language

## Java Versions

Here are a brief history of all the Java versions with its release date.

<b>Java Versions</b>	<b>Release Date</b>
JDK Alpha and Beta	1995
JDK 1.0	23rd Jan 1996
JDK 1.1	19th Feb 1997
J2SE 1.2	8th Dec 1998
J2SE 1.3	8th May 2000
J2SE 1.4	6th Feb 2002
J2SE 5.0	30th Sep 2004
Java SE 6	11th Dec 2006
Java SE 7	28th July 2011
Java SE 8/1.8	18th Mar 2014
Java SE 9	21st Sep 2017
Java SE 10	20th Mar 2018
JAVA SE 11	25th Sep 2018
JAVA SE 12	19th Mar 2019
JAVA SE 13	17th Sep 2019
JAVA SE 14	17th Mar 2020
JAVA SE 15	15th Sep 2020 (latest Java Version)

**JDK :-** JDK stands for java development kit, it internally contain JRE and JVM where JRE stands for java runtime environment and JVM stands for java virtual machine, JDK provide all the tools to work with java language (JDK = JRE + JVM)

**JRE :** JRE stands for java runtime environment, it provide environment to execute the java program, it internally contain JVM which is responsible to execute java program (JRE = JVM + libraries )

**JVM :** JVM stands for java virtual machine, it is the software in the form of **interpreter** written in C language through which we can execute our java program.

After compiling .class file is generated and then interpreter further process it and gives the output

Compiler – Check the complete program at a time

Interpreter – Run the program line by line

### Features of Java

1. Object oriented
2. Platform independent
3. Simple language (no pointer and operator overloading)
4. Secure (java member are private and virus free)
5. Portable :- compile source code and generate bytecode then bytecode can run on any system
6. Compiled and interpreted :- First compiler convert .java file (source code) to .class file (bytecode) then interpreter (JVM) convert .class file to machine code and then execute it.
7. Robust (inbuilt exception handling, automatic garbage collection)
8. Distributed (can run on internet as application can be distributed, uses internet protocol like HTTP, FTP )
9. Multi-threaded (parallel execution)
10. Performance (bytecode execute very fast)
11. Dynamic (can add things runtime)
12. Popular (free and easy)

**Purpose** :- Write once run anywhere, platform independent

.class file is of size 8 bits so said to be bytecode

### Compilation process

.java (source code) ---> compiler (javac) ---> .class (size 8 bits) it is in bytecode 0101

.class ---> interpreter (JVM – class loader) ---> machine language -> output

### Program: Basic Java Program

#### Java 1st program:

1. Create java project
2. create java package
3. create java class
4. main method
5. printing statement
6. save program (Control + s)
7. run program (click green button)
8. check output--> Console tab

Step1: create java project--> file--> new--> java project--> enter project name--> finish

Step2: create java package--> right click on project name/Src folder-->new--> package--> enter package name--> finish

Step3: create java class-->right click on package -->new-->class-->enter class name--> finish

Step4: create main method

Step5: create printing statement

1 java project--> multiple packages

1 packages--> multiple classes

1 class--> multiple method--> main method

1 main method -->multiple printing statements-->to print messages

### Shortcuts in eclipse:

1. main method: "main"+ control + space
2. Printing statement: "syso"+control + space
3. Run java program:- F11
4. Save java class: Control + "S"
5. undo:- Control + "Z"
6. Select all: Control + "A"
7. delete 1 statement: Control + "d"

How to open Console: Windows-->show view--> Console

How to open Project/Package Explorer: Windows-->show view--> Project/Package Explorer

How to open open perspective: Windows-->perspective--> open perspective--> other

**Public** :- cause we should able to access program from outside class as JVM is going to access it from outside class

**Static** :- able to call main without object, main get automatically called no need to depend on object of class

### Run the java program on terminal / cmd:

- a) Save -> class\_name.java
- b) Compilation -> javac class\_name.java
- c) Execution -> java class\_name

### Variables:

Variables are nothing but piece of memory use to store information. One variable can store one information at a time.

Variables also used in information reusability.

To utilise variables in java programming language we need to follow below steps:

1. Variable declaration (Allocating/Reserving memory)
2. Variable Initialization(Assigning or Inserting value)
3. Variable Usage

**Note**:- According to all programming language dealing with information directly is not a good practice, to overcome this variables are introduced.

### Program :

### Class:

Class is nothing but definition block where programmer can declare

1. Variables / data members
2. Methods / member functions
3. Constructor
4. blocks



## Data Types:

Data type are used to represent type of data or information which we are going to use in our java program.

In java programming it is mandatory to declare datatype before declaration of variable.

In java datatypes are classified into two types :

1. Primitive datatype.
2. Non-primitive datatype.

### 1. Primitive Data Type

There are 8 types of primitive data type

- All primitive data type are known as **keywords**
- Memory size of primitive data type is fixed
- It start with lower case

#### Syntax:

Datatype variable\_name;    int num1;

#### The types of primitive datatype are:

(Numeric + Non-decimal):- 10, 99, 100

Ex: 80,85,10,5..etc

**byte** – Value range from -128 to +127

Takes 1 byte (8 bit)

Default value is zero

Ex. byte b = 10;

**short** – Value range from  $-(2^{16})/2$  to  $+(2^{16})/2 - 1$

Takes 2 byte

Default value is zero

Ex. short d = 20;

**int** – Value range from  $-(2^{32})/2$  to  $+(2^{32})/2 - 1$

Takes 4 byte

Default value is zero

Ex. int e = 100;

**long** – Value range from  $-(2^{64})/2$  to  $+(2^{64})/2 - 1$  (add l/L at last by default)

Takes 8 byte

Default value is zero

Ex. long h = 1000l OR 1000L;

(Numeric + decimal):-

Ex: 22.5, 22.8, 6.4....

**float** – Value range from  $3.40282347 \times 10^{38}$ ,  $1.40239846 \times 10^{-45}$

Takes 4 byte

Default value is 0.0f

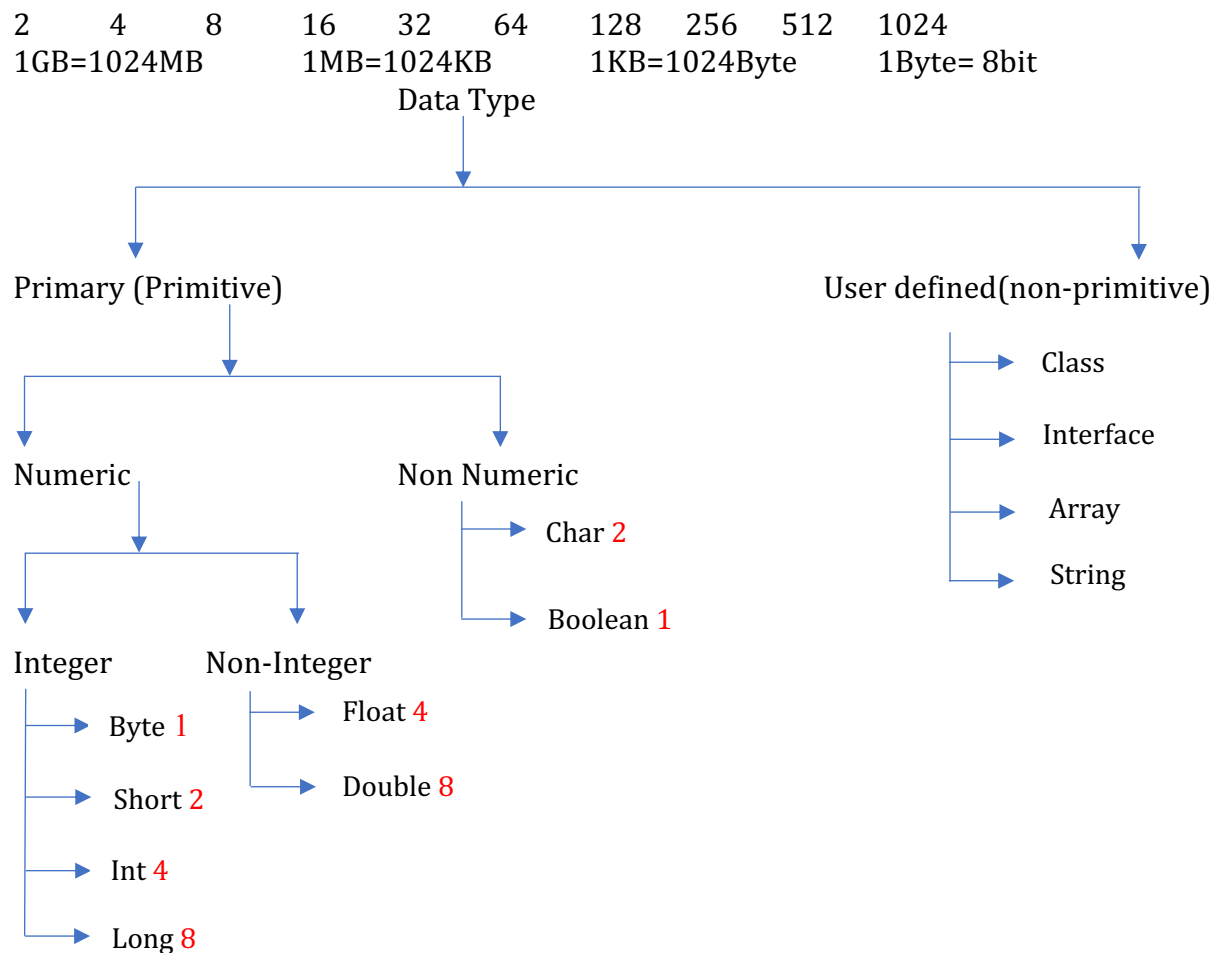
Ex. float f = 10.10f;

**double** – range from  $1.7976931348623157 \times 10^{308}$ ,  $4.9406564584124654 \times 10^{-324}$   
(by default decimal value is of type double)

Takes 8 byte  
 Default value is 0.0d  
 Ex. double g = 122.1;

**char** – Value range from 0 to 65535 ( $2^{32} - 1$ )  
 Takes 2 byte (because it support Unicode)  
 Default value is '00000'  
 Ex. 'a', 'B', '0' etc.  
 Ex. char c = '+';  
 1Kg = 1000gram    1MB =1024 kb. Binary

**Boolean** – Value can be true or false  
 Size depend on JVM  
 Default value is false  
 Ex. true, false  
 boolean a = true



Data Type	Default Value (for fields)
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d

char	'\u0000'
String (or any object)	null
boolean	false

There are four types of non-primitive data type

- |       |           |       |        |
|-------|-----------|-------|--------|
| Class | Interface | Array | String |
|-------|-----------|-------|--------|
- It is known as identifier
  - Memory size is not fixed
  - Memory allocated at run time

## Program:

### Identifier

Identifiers are the name given to variables, classes, methods, package etc

Rules for defining identifier :-

- The only allowed character for identifier are alpha numeric ex. [a-z] [A-Z] [0-9] [\$ & \_]
- It should not start with number or digit  
Allowed :- abc, abc\_123, abc\$123  
Not allowed :- abc@123, 123\_abc
- Identifier are case sensitive
- Reserved word cannot be used as identifier
- White spaces are not allowed

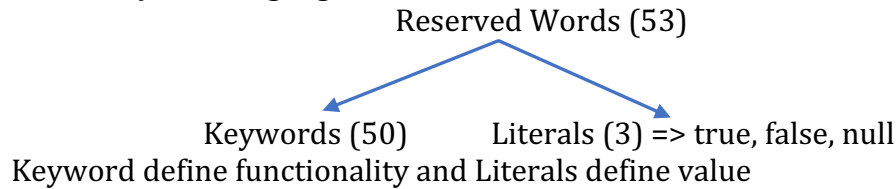
### Java Keywords:

Keywords are predefined, reserved words used in Java programming that have special meanings to the compiler. For example:

abstract	goto	synchroniz
assert	if	ed
boolean	implements	this
break	import	throw
byte	instanceof	throws
case	int	transient
catch	interface	try
char	long	void
class	native	volatile
const	new	while
continue	package	
default	private	
do	protected	
double	public	
else	return	
enum	short	
extends	static	
final	strictfp	
finally	super	
float	switch	
for		

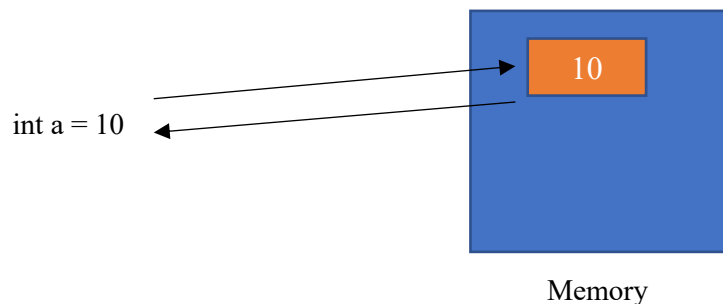
## Reserved Words

Any programming language reserved some words to represent functionality defined by that language, these words are called as reserved words



## Variable

- It is name of memory location.
- It is user defined given by user.
- Variable can store any type of value.
- One variable can store one information at a time and it can be change.



## Control statements

It is used to control the flow of execution of program based on certain condition

1. If statement
2. If else statement
3. Else if statement
4. Nested if statement
5. Switch statement

### 1. If statement

It is used to decide whether certain statement or block of statement will be executed or not i.e. if the condition is true then only

**Program:**

### 2. If else statement

Else statement execute a block of code when condition is false

**Program:**

### 3. Else if statement

Here user can decide multiple option while executing the statement from top as soon as one of the condition is true the statement associated with will execute and rest of condition are skipped. If none of the condition is true then final else statement will execute

Note :- If multiple condition are true then it will execute first condition

## Program:

### 4. Nested if statement

It means if statement inside if statement

## Program:

### 5. Switch Case

The Java switch statement executes one statement from multiple conditions.

Points about Switch Statement:

There can be one or N number of case values for a switch expression.

The case value must be of switch expression type only.

The case values must be unique. In case of duplicate value, it renders compile-time error.

Each case statement can have a break statement which is optional.

The case value can have a default label which is optional.

*Switch Case Syntax :*

```
switch(expression){ case value1:
//code to be executed;
break; //optional case value2:
//code to be executed;
break; //optional .....
default:
code to be executed if all cases are not matched;
}
```

## Program:

## Program :

## Program :

**Loops :** Loops are used to execute a set of instructions/ methods repeatedly when some conditions become true.

Advantages:

- Fast execution
- Reusability
- Decrease line of code
- Less memory required

In Java we have four Loops:

- For loop
- Do-while loop
- While loop
- For each loop – Array, Collection

For loop is used to iterate a part of program multiple times if number of iteration is fixed

While loop is used to iterate a part of program multiple times if number of iteration is not fixed

Do-While loop is used to iterate a part of program multiple times if number of iteration is not fixed and you must have execute loop at least once.

**For Loop:** We can initialize the variable, check condition and increment/decrement value in one line.

Syntax :

```
for(initialization; condition; incr/decr){ //statement or code to be executed
}
```

```
for (statement 1; statement 2; statement 3)
{
    // code block to be executed
}
```

Statement 1 is executed (one time) before the execution of the code block.

Statement 2 defines the condition for executing the code block.

Statement 3 is executed (every time) after the code block has been executed.

**Program :**

```
public class SecondProgram {
    public static void main(String [] args){
        for (int i=0;i<10;i++){
            System.out.println(i);
        }
    }
}
```

**Program :- Print number from 1 to 5**

**Program :- Print number from 55-75**

**Program :- Print even number from 2-10**

**Program :- Print odd number from 5-51**

**Program :- Print number 5 to 1**

**Program :- Print even number from 100 to 2**

**Program :- Print odd number from 9 to 1**

**Program :- Print cube of values from 1 to 5**

**While Loop:**

while loop is preferred to use, If the number of iteration is not fixed. It is pre-test loop it is also called as entry control loop

```
Syntax : while(condition){
    //code to be executed
}
```

**Program :**

## Do-While loop

do-while Loop: If the number of iteration is not fixed and **you must have to execute the loop at least once**, it is recommended to use do-while loop. It is post test loop

Syntax :

```
do{
//code to be executed
}
while(condition);
```

## Program :

### Class

It is user defined prototype from which objects are created. It represent set of method that are common to all objects of one type

- Collection of objects is called class.
- A class can also be defined as a blueprint from which you can create an individual object.

### Object

- It is basic unit of object oriented programming and represent real life entities. A typical java program create many object used to interact or call the method
- Any entity that has state and behavior is known as an object. For example a chair, pen, table, keyboard, bike, etc. It can be physical or logical.
- Example: A man is an object because it has states like name, height, weight, etc. as well as behaviors like walking, running etc.

Characteristics of objects are:

1. State
2. Behaviour

eg. Marker

state- colour, size, price, weight

behaviour- write, throw

eg. Car

state: What the objects have: Speed, Gear, Direction, Fuel level, Engine temperature

Behaviours: Change Gear, Go faster/slower, Go in reverse, Stop, Shut-off

Student:

State: what the objects have, Student have a first name, last name, age, etc

Behaviour: what the objects do, Student attend a course "Software testing"

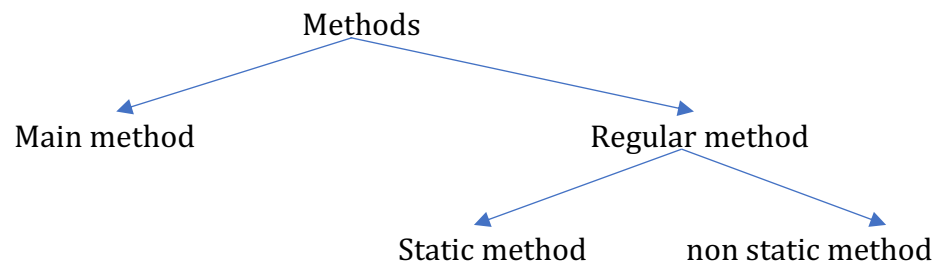
In the filed of java each and everything is consider to be as object.

**Object is copy of class or instance class which is having state & behaviour.**

where state stands for variable(data member) & behaviour stands for methods(member functions).

### Methods:

A method is a block of code which only runs when it is called.  
Methods are used to perform certain actions, and they are also known as functions.  
You can pass data, known as parameters, into a method.



Why to use method:

- To reuse the code
- At the time of program execution main method is going to execute automatically where as regular method has to be called
- At the time of program execution priority is scheduled for main method only
- To execute regular method programmer need to make call from main method until and unless call is not made regular method will not get executed

### 1. Main method (pre-defined)

In any Java program, the main() method is the starting point from where compiler starts program execution.

So, the compiler needs to call the main() method.  
without main method we can't run any java program.

### 2. Regular method (user defined)

**Static Method** : If you apply static keyword with any method, it is known as static method.

- A static method belongs to the class rather than the object of a class.
- A static method can be invoked (call) without the need for creating an instance of a class. (object of class)
- A static method can access static data member and can change the value of it.

**Non-Static Method** : If you do not apply static keyword with any method, it is known as non-static method.

- A non-static method belongs to the object of the class.
- A non-static method can be invoked (call) with an instance of a class (object).
- A non-static method can access static data member and method without creating instance of class.

#### 1. Static method call from same class

When we used void in method declaration, method cannot return any value

**Program :**

#### 2. Static method call from different class

**Program :**

#### 3. Static method with parameter in same class



## **Program :**

### **4. Static method with parameter from different class**

## **Program :**

### **5. Static method with return type in same class**

For method to return a value we need to use relevant data type such as int, float, String instead of void and use return keyword inside the method

- Information can be passed to method as parameter
- Parameter act as variable inside method
- It is specified after method name inside parenthesis
- We can add as many parameter as we need

## **Program:**

### **6. Static method with return type from different class**

## **Program :**

### **7. Non static method call from same class**

## **Program12:**

### **8. Non static method call from different class**

## **Program13:**

### **9. Non-Static method with parameter in same class**

## **Program14:**

### **10.Non-Static method with parameter from different class**

## **Program15:**

### **11.Non-Static method with return type from same class**

## **Program16:**

### **12.Non-Static method with return type from different class**

## **Program17:**

### **13.Static method and non-static method call from same and different class with parameter, without parameter and return type**

## **Program18: Assignment**

## **Constructor**

- Constructor is a special member of class
- It is used to initialize data member (objects/variables) of class and to load non-static member of class into object
- According to java language each and every class should have constructor
- **The constructor is called when an object of a class is created.**

## **Rules**

- Constructor name should be same as class

- It does not have any return type
- Any number of constructor can be declared inside java class but constructor name should be same as class name while argument must be different

#### Use of Constructor

- To copy/load all members of class into object --> when we create object of class
- To initialize data member/variable

Constructor are classified into two types

##### 1. Default constructor

- If constructor is not declared in java class then at the time of compilation compiler will provide constructor for class called as default constructor
- If programmer has declared the constructor then compiler will not provide default constructor
- The Constructor provided by compiler at the time of compilation is known as Default Constructor

##### 2. User defined constructor

If programmer declaring constructor in java class then it is considered to be user defined constructor

User defined Constructor are classified into 2 types

1. Without/zero parameter constructor

// example-without parameter constructor --ex. addition

2. With parameter constructor

// example-with parameter constructor- only 1 constructor – ex. addition

// example-with parameter constructor- multiple constructor -- ex. addition

##### a) Zero argument or zero parameter

**Program:**

##### b) Argument or parameterized constructor

**Program:**

#### Static block

It is executed at the time of loading .class file in jvm memory

#### Instance block

It is like to method which has no name, we can write instance block inside class not method.

- It executed before constructor.
- Variable can be declare inside instance block not method
- Time consuming code can be written in instance block
- **To execute instance block object should be declared**

**Program:**

## Types of variable

- a) Local
- b) Instance/Global
- c) Static

**a) Local variable** :- A variable which declare inside method/block/constructor called as local variable

Scope of local variable remains only within the method/block/constructor.

Local variable is also called temporary variable.

### Program :

**b) Instance/Global variable**:-A variable which declare inside the class but outside of all method/block/constructor is called instance variable

scope of global variable remains thought the class.

global variable is also called permanent variable.

**c) Static/ Class variable** :- A variable which declare with the help of static keyword is called as static variable.

1. static variable call from same class --> variableName

2. static variable call from diff class--> className.variableName

Note: we can access static global variable in both static & non-static method

## Object-Oriented Programming System(OOPS)

Java programming language is very popular in software industry because of OOPS concept.

OOPS concept provides 5 important pillar /principles for the language they are

- 1. Inheritance
- 2. Polymorphism

- 3. Encapsulation
- 4. Interface
- 5. Abstraction

## Inheritance

It is one of the object oriented principal where one class acquire properties of another class with the help of **extends** keyword

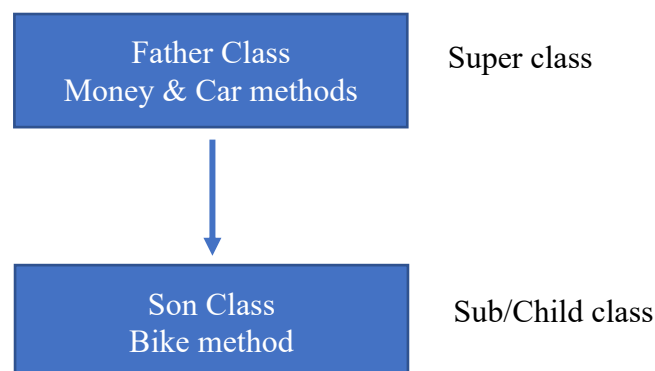
- The class from where properties are acquired is known as super/parent class
- The class to where properties are inherited are known as sub/child class
- Inheritance takes place between 2 or more than 2 classes.

It is classified into four types

1. Single level
2. Multi-level
3. Multiple
4. Hierarchical

### 1. Single level inheritance

- It is an operation where inheritance takes place between two classes
- To perform single level inheritance two classes are mandatory



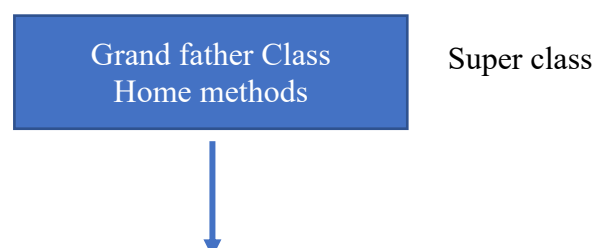
**Program:**

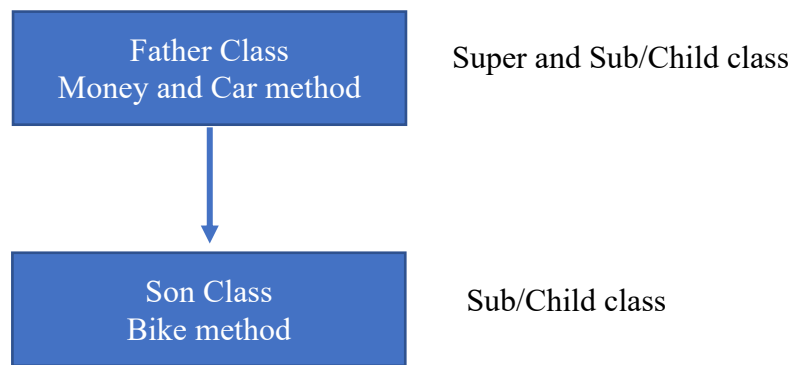
**Program:**

### 2. Multi-level

It takes place between three or more classes

One sub class acquires the properties of another super class and that class acquire properties of another super class and phenomena continues





### Program:

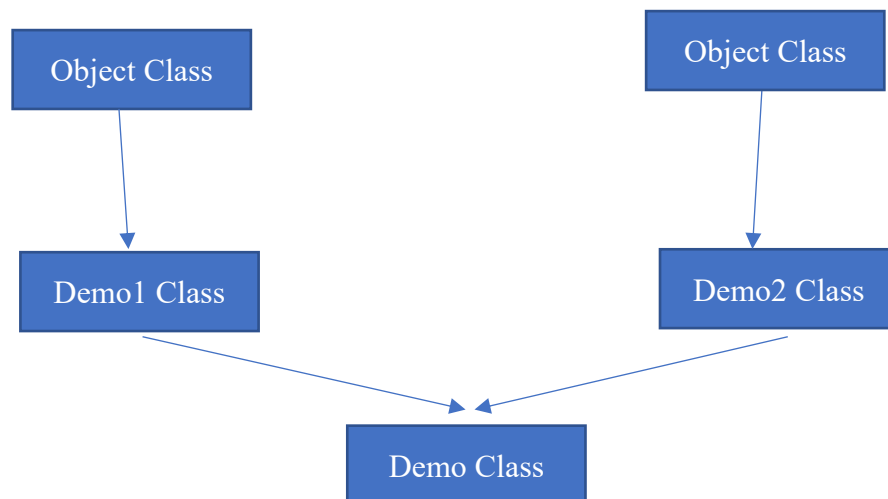
#### 3. Multiple Inheritance

One sub class acquire the properties of two super classes at same time is known as multiple inheritance

But java does not support it due to ambiguity problem

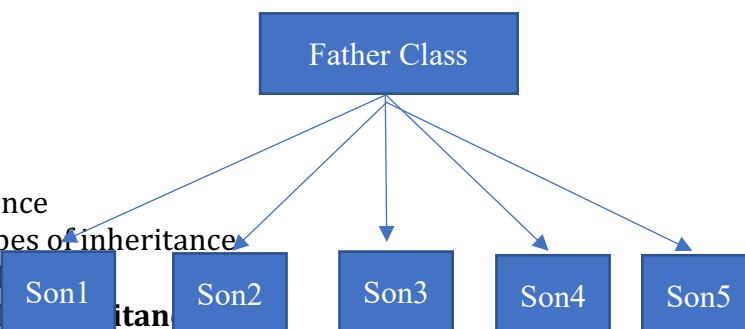
**Note: object class is the super most class in java**

By using interface we can achieve Multiple Inheritance.



#### 4. Hierarchical Inheritance

Two or more than two sub classes acquires properties of one super class  
multiple sub classes can acquire properties of 1 super class is known as hierarchical Inheritance.



### Program25:

What is inheritance

What are the types of inheritance

What is object class

**What is multiple inheritance**

Why we can not implement multiple inheritance in java using classes?

## This Keyword

Every class have unique reference number, so reference variable (object) refer the unique reference number of class also this keyword refer the unique reference number.

- This keyword refer to current object inside the method or constructor
- Whenever the name of instance variable and local variable are same then JVM confused which one is local variable and which is instance variable so to avoid this we should used this keyword
- It is also used when we want to call zero parameter constructor of its own class
- It is also used to call parameterized constructor of its own class

**Program :**

**Program :**

**Program :**

**Program :**

## Super Keyword

- Super keywords refer to the objects of super class
- It is used when we want to call super class variable, method and constructor through sub class object.
- Whenever super class and sub class variable and method name are same then JVM confused which one to call so to avoid this we should use super keyword

It is used to access global variable from super class or different class

**Program :**

**Program :**

**Program :**

**Program :**

**Program :**

## Access Specifier/Modifier

It is used to represent scope of member of class (variables, methods, constructor).

- 1) Private :- If we define any member of class as private then scope of that member remain only within the class. It cannot access from other classes
- 2) Default :- In this scope of member remain only within the package. There is no keyword to declare method. It can't be access from other packages.
- 3) Protected :- in this scope of that member remain within package only. The class which is present outside package can also access it by on condition i.e. inheritance operation
- 4) Public :- Scope of these member remain throughout the project

Sr. No		Private	Public	Protected	Default
1	Same Class	YES	YES	YES	YES
2	Sub class in same package	NO	YES	YES	YES
3	Non sub class in same package	NO	YES	YES	YES
4	Sub class in another package	NO	YES	YES	NO
5	Non sub class in another package	NO	YES	NO	NO

### **Program29:**

**Package 1 => Class A {private int a}**

**Package 1 => Class B extends A { }**

**Package 1 => Class C**

**Package 2 => Class D extends A { }**

**Package 2 => Class E { }**

## **Polymorphism**

It is one of the OOPs principle where one object showing different behaviour at different stages of life cycle.

Polymorphism is an latin word where poly stand for many & morphism stands for forms.

It means having many forms OR ability of message to be displayed in more than one form

Ex. A person can be father, husband, employee, friend, trainer

It allowed single action in different way

Poly => many and morphs = form

In java it is classified into two types

- 1) Compile time polymorphism
- 2) Run time polymorphism

### **1) Compile time polymorphism**

Which exist in compile time, it is also called as early binding OR static polymorphism

In Compile time Polymorphism method declaration is going to get binded to its definition at compilation time, based on argument/input/parameter/signature is known as compile time Polymorphism.

As binding takes during compilation time only, so it is also known as early binding.

Method overloading is an example of compile time Polymorphism.

## **Method overloading**

When a class contain more than one method with same name and different argument/parameter/inputs is called as method overloading

### **Program34:**

### **2) Run time polymorphism**

In Runtime Polymorphism, method calling is going to get binded to its definition at Runtime/execution time, based on object creation is known as runtime Polymorphism.

As binding takes during Runtime/execution time, so it is also known as late binding.

//once binding is done, again rebinding can be done, so it is called dynamic binding.

Method overriding is an example of Runtime Polymorphism.

## **Method overriding**

When we write method in super and sub class in such a way that method name and parameter must be same then it is known as method overriding

Acquiring super class method into sub class with the help of extends keyword & changing implementation/definition, according to subclass specification is called method overriding

**Program:**

**Program:**

**Program:**

Compile Time	Run Time
1) Exist at the time of compilation	1) Exist at run time
2) Early binding	2) Late binding
3) Method overloading	3) Method overriding
4) Static polymorphism	4) Dynamic method dispatch

### Encapsulation

It is mechanism where we can wrap data member and member method in single unit (class)

Note:- declare class member as private and method should be public

**Program: Data is new oil**

### Abstraction

abstraction is one of the oops principle in java.

It is the process of hiding implementation details from user only highlighted services are visible to user

The scenario of Abstraction is "if customer is visiting or making use of any application, then he should utilize functionality only & he should not feel any backend code processing"

Advantage :- Security

Enhancement

Implementation of Abstraction

Abstract class

Interface

### Abstract class

- A class declare with abstract keyword is known as abstract class
- It may or may not contain abstract method
- To inherit abstract class we need to inherit it with sub class
- If class contain partial implementation then we should declare it as abstract
- programmer can declare incomplete methods as abstract method, by declaring keyword called "abstract" in front of method.
- It is incomplete class where programmer can declare complete and incomplete method
- An incomplete method is method where only method declaration is present and method definition is not available



- We cannot create object of abstract class
- To create object of abstract class we must have concret class (sub / child class)
- Incomplete method is said to be abstract method

**Program :**

**Program :**

### **Concret Class**

A class which provide definition for all incomplete method present inside abstract class with the help of extends keyword is known as concret class

*can 1 Abstract Class extends another Abstract Class ? yes*

### **Abstract method**

A method which contain abstract modifier at the time of declaration is called abstract method

Note:- It can only be used in abstract class

It does not contain any body, contain declaration only

Abstract method must be overridden in sub class else sub class will also become abstract

When action is common but implementation is different then we should use abstract class

**Program: Assignment – RBI (method IR) ICICI HDFC SBI**

**Program:**

### **Interface**

Interface is just like class which contain only abstract (incomplete) method

To achieve interface java provide a keyword called **implements**

Interface method are by default public and abstract

Interface variable are by default public, static, final

Interface method must be overridden in implementation class

Interface deals with client developer

constructor concept is not present inside Interface.

To create object of Interface programmer need to make use of Implementation class using implements keyword.

**Interface support multiple inheritance.**

- It is not 100% abstract in nature
- It is one of the object oriented principal
- There is no constructor principal
- We cannot create object of interface
- To create object we must have implementation class
- Until we complete all method in interface it is said to be abstract class

**Program :**

**Program:**

**Program:**

**Program:**

**Program:**

### **Implementation Class**

A class which provide definition for all incomplete method present in interface with the help of implements keyword is known as implementation class

- **There is no object class for interface**

## **Implementation of multiple inheritance**

### **Program :**

### **Generalization**

Extracting all common important properties from subclass and declare it in super class (interface) and providing implementation according to subclass specification

Generalization file can be interface, abstract class or java class but **interface** is recommended

### **Program:**

## **Type Casting**

**What is casting?**

**What is type casting?**

**What is up casting, what is usage of that in Project? - VIMP**

Converting one type of information into another type is known as typecasting OR

Converting one data type into another data type is called as type casting

It is classified into two types

- 1) Primitive casting
- 2) Class / Non-primitive casting

### **1) Primitive casting**

- Converting one data type information into another is known as primitive casting
- In this compiler take care of conversion operation internally, so it is also called as automatic casting

It is further classified into three types

- a) Implicit casting (automatic)
- b) Explicit casting (forceful)
- c) Boolean casting (incompatible)

### **a) Implicit casting**

Converting lower data type information into higher data type is known as implicit casting

- It is automatically performed by compiler
- External resource support is not allowed or not required
- Ex. int a = 3; (memory size 4 byte)  
System.out.println(a);  
double b = a;  
here we are assigning int type information to double  
system.out.println(b);  
It is also called as widening casting where memory size increases.

### **b) Explicit casting**

Converting higher size of data type into lower size information is known as explicit casting

By default

- External support is required

Ex. double a = 2.5;          8 byte          2.5

int b = (int) a;          4 byte          2

We are converting information then inserting it into respective type

Data loss

### Program33:

#### c) Boolean casting

It is consider to be incompatible type because boolean is unique data type where information is already provided inside it (TRUE/FALSE)

- For Boolean data type programmer cannot provide additional information so it is incompatible.

#### 2) Class / Non-primitive casting

Converting one type of class into other or assigning property of one class into another is known as class casting

It is classified into two types

- a) Up-casting
- b) Down casting

##### a) Up-casting

Assigning sub class properties into super class is known as up-casting

- Before performing upcasting, inheritance operation should takes place
- Then property of super class comes into sub class
- Sub class can have its own properties
- At the time of upcasting, property which are inherit from super class are only eligible for operation, properties of sub class are not eligible for operation

##### b) Down casting

Assigning super class properties into sub class is known as down-casting

- But it is not allowed in java still if we use down casting there will not be any compiler error but runtime exception will occur
- Java.lang.ClassCastException

### Program:

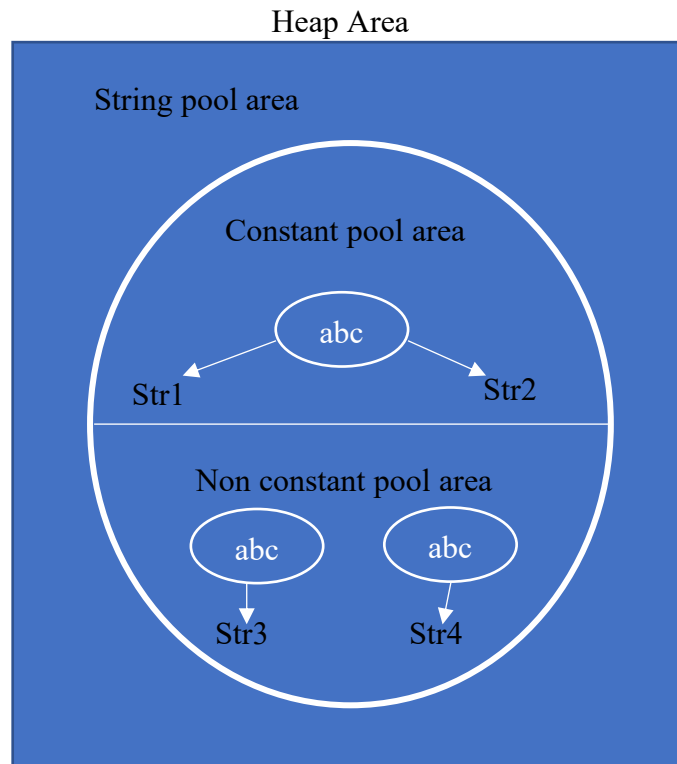
#### String

String class:

1. String is non-primitive data type, memory size is not fixed.
2. String is use to store collection of characters.
3. String is a inbuilt/ready made class present inside "java.lang" package.
4. String class is final class can't be inherited to other classes.
5. At the time of String declaration, initialization, object creation takes place.
6. String objects are immutable in nature/can't be change.
7. Object creation of String can be done in 2 ways:
  1. Without using new keyword

2. Using new keyword

8. String objects are going to get stored inside String pool area which is present inside heap area.



String object is stored inside of string pool area which present inside heap area

Object creation of string can be done in two ways

- a) Using new keyword
- b) Without using new keyword

String str = "xyz"; => stored in constant pool area

String str1 = new String("xyz") => stored in nonconstant pool area

String pool are classified into two types

- a) Constant pool area
- b) Non-constant pool area

**a) Constant pool area**

During object creation if we do not use new keyword then object creation takes place in constant pool area  
Duplicate data is not allowed

**b) Non-constant pool area**

If we use new keyword then object creation takes place in non-constant pool area  
Duplicate data is allowed

**Reason:-**

For string object, string constant pool area concept is available, multiple reference pointing same object. By using one reference if we allowed to change content then remaining references affected to prevent this **immutability** required

Survey :- Name, age, which city you like to live, what about your current city, city=Pune  
1000 people

What is string

Why string is immutable

Can you tell any 5-8 (10) string method, with explanation

**String class methods**

- a) **length()**
- b) toUpperCase
- c) toLowerCase
- d) **equals()**
- e) equalsIgnoreCase
- f) contains
- g) isEmpty()
- h) **charAt(int index)**
- i) indexOf
- j) lastIndexOf
- k) startsWith
- l) endsWith
- m) substring(3)
- n) substring(2,3)
- o) concat
- p) replace
- q) **split**
- r) trim

**Program:**

**Program**

**Program:**

**Program:**

**Program:**

**Program:**

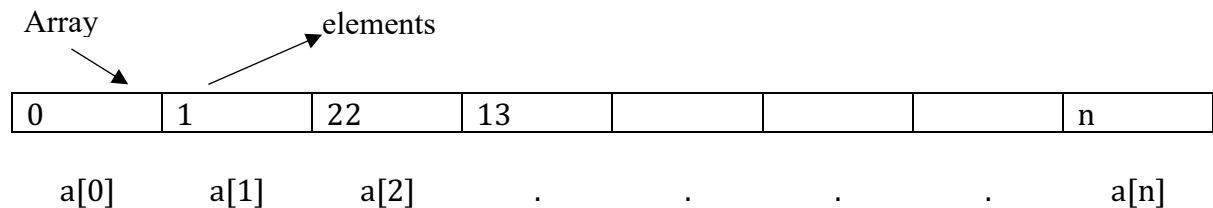
**Program:**

**Array**

Need?

Array is an object in java which contain similar type of data in contiguous memory location

- It is data structure used to store collection of information (huge data)
- It is homogenous in nature (similar type of data can be maintained in one array)
- Array declaration needs to be done with size
- Array cannot be expanded
- It is memory cell starts from zero
- The information inserted in array is consider to be an element



Disadvantage:

- Array has limited length
- Array is homogenous (only one data type)

Syntax:

data\_type [] variable\_name;      OR      data\_type var\_name[];

String ar[] = new String [5];

ar[0] = "Andy";  
ar[1] = "Anna";  
ar[2] = "Jon";  
ar[3] = "Ron";  
ar[4] = "Harry";

} Initialization

OR

String ar[] = {"Andy","Anna","Jon","Ron","Harry"}

**Program:**

**Program:**

**Program:**

**Program:**

**Program:**

**Final** : final keyword in java is used to restrict the user. Variable, Method and Class can be defined final.

- Final Variable : Value of Final Variable can't be change, it is constant.
- Final Method : Final Method can't be overridden in any child class.
- Final Class : Final Class can't be extend in java.

**Program**

## Exception Handling

### Exception

During the execution of java program JVM faces abnormal situation base on code declaration then it trigger an event said to be exception which result in termination of java program then code is consider to be non-feasible code for execution

If any event is generated by JVM then programmer need to handle event so that's all line in program get executed.

An exception is unexpected/unwanted/ abnormal situation that occur at run time is called exception

Ex. if power failure at our home (power cut exception - power backup),

medical issue with us (ill exception – by medicine or doctor ),

if we try to read file in java but not getting the file is HDD (display the message)

**Program:**

## Exception Hierarchy

Throwable is a super class of java exception hierarchy which contain two sub classes

- a) Exception
  - a. RuntimeException (Unchecked)
  - b. IOException (Checked)
  - c. SQLException (Checked)
  - d. InterruptedException (Checked)
  - e. ClassNotFoundException (Checked)
- b) Error (Unchecked)
  - a. StackOverflowError
  - b. OutOfMemoryError
  - c. IOError
  - d. LinkageError
- a. RuntimeException (Unchecked)
  - a. ArithmeticException
  - b. NullPointerException
  - c. NumberFormatException
  - d. IndexOutOfBoundsException
    - i. ArrayIndexOutOfBoundsException
    - ii. StringIndexOutOfBoundsException
- b. IOException (Checked)
  - a. EOFException
  - b. FileNotFoundException

## ***Mechanism to handle the exception***

Following are the technique to handle the exception

- a) Try
- b) Catch
- c) Throw
- d) Throws
- e) Finally

To handle predefined exception created by user and we are taking the responsibility of it then we can use try-catch block

## **Throw**

If we want to create our own exception object and wants to throw the exception as we are not taking the responsibility of it and we want JVM should handle that exception then we use throw

## **Throws:**

We can assign exception handling responsibility to caller (JVM or another method)

Needed for checked exception, for unchecked no use of it

No prevention of abnormal termination

Try Block :- Put risky code in try block

- Used to declare risky code only
- Controller visit try only once throughout lifeline of program
- Try block followed by catch or finally block
- Multiple try block are not allowed

Catch Block :- it handle the exception throws by try block

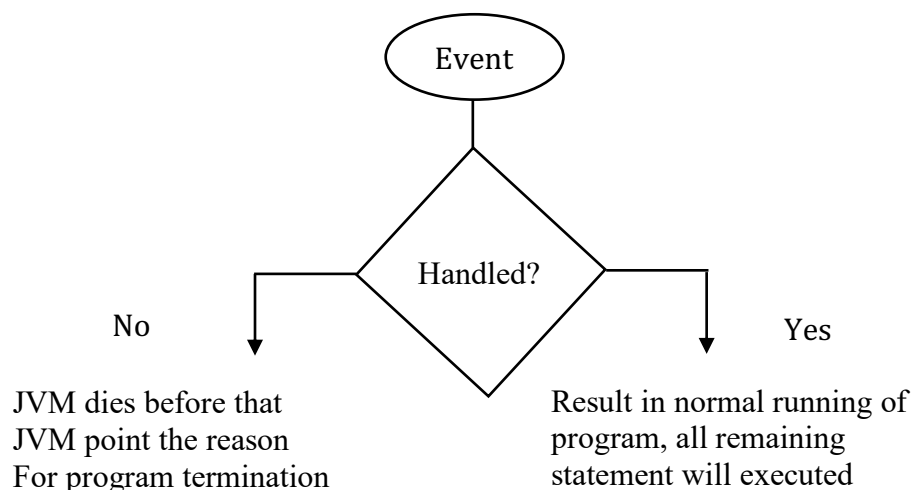
- Used to handle event generated from try block
- **Executed if event generated in try block**
- Should be declare after try block
- Any number of catch block can be declare for single try

Note :- Catch will not execute if no exception in try block

Finally Block is followed by try block and always executed whenever the exception occurs or not.

Finally Block is used to execute important code such as closing connection, stream, cleanup code etc.

### Program:



To handle the event / exception below block are used

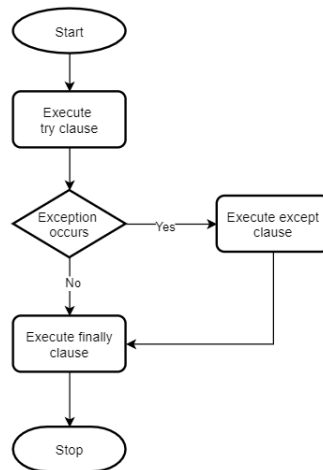
Syntax:

```

try {
// Risky code
}
catch(event / exception name ref_variable) {
// event handled message
}
  
```

### Program:





```

try {
statement1;
statement2;
statement3;
}
catch(Exception e) {
statement4;
}
finally {
statement5;
}
statement6;

```

Case 1 : No exception => 1 2 3 5 6 (normal execution)

Case 2 : Exception at 2 and match with catch => 1 4 5 6 (normal execution)

Case 3 : Exception at 2 and not match with catch => 1 5 (abnormal termination)

Case 4 : Exception at 2 and even in catch => 1 5 (abnormal termination)

Case 5 : Exception at 2 and even in finally => 1 (abnormal termination)

### Program:

**Program :- Multiple try catch**

**Program : Multiple catch block**

Checked Exception	Unchecked Exception
Exception checked by compiler for smooth execution of program	Exception not checked by compiler and handled by JVM
Mostly occurred so compiler checked	Rarely occurred so compiler not take care
Ex. file not found, InterruptedException	Ex. Arithmetic, null pointer, arrayIndexoutofBound

**Final, finally, finalize**

## **Finalize**

Method available in object super class

Release the resources allocated by unused object, before removing unused object by garbage collection

Finalize is protected by default/ but we can use public

### **Program:**

## **What is garbage collection**

Delete the used entity

Data Base connection and network connection

Then finalize method is to be called by garbage collection before destroying the object for object to perform cleanup activity

<b>Finally</b>	<b>Finalize()</b>
It is block used in try catch	It is method
Used to close the resources open in try block	Used to take the resource back from unused object, garbage collector deallocate the resource before destroying the object