**Java**

1. **Installation**
   1. **JDK**
   2. **JRE**
   3. **JVM**
   4. **lib**
   5. **bin**
2. **Editor – Eclipse**
3. **OOPS Concepts**
   1. **abstraction**
   2. **encapsulation**
   3. **polymorphism**
   4. **inherence**
4. **Class**
   1. **components of Java Programming**
   2. **Standards**
5. **Packages**
6. **Data types**
   1. **primitive data types**
   2. **derived data types**
7. **Variables**
   1. **local variables**
   2. **global variables**
      1. **instance variables**
      2. **static variables**
      3. **constants**
8. **methods**
   1. **methods without arguments**
   2. **methods with arguments**
   3. **methods without returntype**
   4. **methods with returntype**
   5. **static methods**
   6. **non-static methods**
   7. **final methods**
   8. **Overloading and Overriding**
9. **Access Specifiers**
   1. **private**
   2. **default**
   3. **protected**
   4. **public**
10. **Access modifiers / Non functional Access specifiers**
    1. **static**
    2. **abstract**
    3. **final**
    4. **synchronized**
11. **Looping and Conditional Statements**
    1. **for**
    2. **while**
    3. **do,while**
    4. **foreach**
    5. **if**
    6. **if,else**
    7. **if,elseif,else**
    8. **switch**
12. **Operators**
    1. **arithmetic**
    2. **logical**
    3. **conditional**
    4. **unique**
    5. **ternary**
13. **Constructors**
    1. **default constructor**
    2. **parameterized constructor**
14. **String**
    1. **string functions**
    2. **important java – programs**
    3. **StringBuffer**
    4. **StringBuilder**
15. **Exception Handling**
    1. **try**
    2. **catch**
    3. **throw**
    4. **throws**
    5. **finally**
    6. **Userdefined Exception**
16. **File Handling**
    1. **read – properties file, text file, Excel file**
    2. **write– properties file, text file, Excel file**
    3. **create files /folders**
17. **Collections**
    1. **Iterator**
       1. **next**
       2. **hastNext**
    2. **List**
       1. **ArrayList**
       2. **LinkedList**
       3. **Vector**
    3. **Set**
       1. **HashSet**
       2. **TreeSet**
       3. **LinkedHashSet**
    4. **Map**
       1. **HashMap**
       2. **TreeMap**
       3. **LinkedHashMap**
    5. **Generics – Parameterized Collections**
18. **Wrapper Classes in Java**

**Selenium Automation**

1. **Some intro to Testing**

* **What is Software Testing**
* **What are the different type of testing we do ???**
* **What are the drawbacks of manual testing**
* **Why Automation is needed ?**
* **What are the Tools Available in Market ?**
* **Why Selenium ????**

1. **Features Of Selenium**
2. **History**
3. **Versions**

* **Selenium IDE**
  + **Beginners – NOT much time**
  + **Object Identification mechanism** 
    - **CSS**
    - **xpath (15)**
      * **basic**
      * **using function** 
        + **text()**
        + **contains()**
        + **starts-with()**
      * **using Regular expression** 
        + **AND**
        + **OR**
        + **NOT**
      * **Traversing from patent to child**
      * **Traversing from child to parent**
      * **using axes function**
        + **following**
        + **preceding**
        + **following-sibling**
        + **preceding-sibling**
        + **parent**
        + **child**
        + **ancestor**
* **~~Selenium RC~~**
* **Selenium WebDriver - 3.141.59**
  + **Operations**
    - **click – button, dropdown, image, hyperlink ...**
    - **type – textbox, text area..**
    - **dropdown elements**
    - **getting text from application**
    - **getting attribute value of a element**
  + **KB and Mouse operations**
  + **Automate –** 
    - **tooltip**
    - **suggestions**
    - **mouse hover**
  + **Browser operation**
  + **End to end tests**
  + **Popup**
    - **alert**
    - **confirmation popup**
    - **hidden division popup**
    - **file download**
    - **file upload**
    - **pageload popup**
  + **Sikuli / Auto IT**
  + **Data driven testing**
    - **Global Data – Common to all the tests**
    - **Local Data- specific to tests**
* **Selenium Grid**
  + **Distributed Execution**

**Frameworks**

* **TestNG**
  + **annotations**
  + **group**
  + **data driven testing**
  + **priority**
  + **parallel execution**
  + **HTML Reports**
  + **execute the failed tests**
  + **customized reports**
* **POM – Page Object Model**

**Maven – Build Automation Tool**

1. **Installation**
2. **Maven build life cycle**
3. **phases of maven**
4. **Create a Maven Project**
5. **batch run – without opening eclipse**

**BDD – Behavioural Driven Development**

1. **Developer, tester, manager, Business analyst, Customer, Architect**
2. **Gherkin language**
   1. **Feature File**
   2. **Step Definition**
   3. **Test Runner**

**GIT – Distributed Version Control Software**

1. **Installation**
2. **Adding roles to the users etc**
3. **Difference between local repository / Global Repository**
4. **Basic GIT Commands**
5. **stash / unstash**
6. **revert the commit**
7. **resolve conflicts**

**Jenkins – CI/CD/CT**

1. **Installation**
2. **Configuration**
3. **Plugin management**
4. **How to create Builds / Jobs**
5. **How to execute the tests**
6. **How to execute the tests Automatically**
7. **How to analyze logs / Reports**

**============================================== JAVA ==================================**

1. **JAVA Installation**

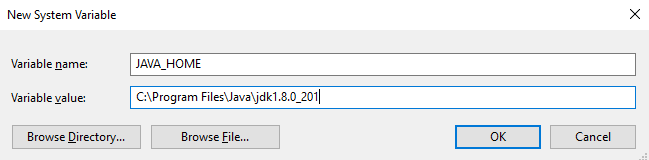
* Go to google.com
* search for download jdk1.8
* [**https://www.oracle.com/in/java/technologies/javase/javase-jdk8-downloads.html**](https://www.oracle.com/in/java/technologies/javase/javase-jdk8-downloads.html)
* Accept the licence agreement
* Create Oracle account
* login to oracle website (download site) and download the software
* double click on the exe file, and follow the instructions
* by default software will be installed on c:\program files\java

1. **Components of JAVA**

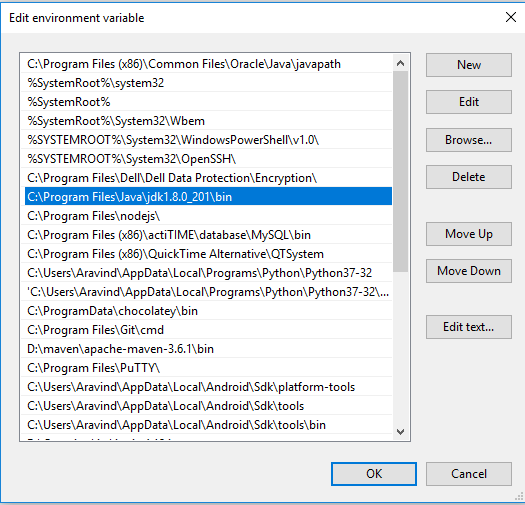
* jdk
  + bin – *executables / commands ( javac, java, javadoc )*
  + lib – jar files / libraries
  + jre
    - bin – dll files
    - lib – files and folders, rt.jar
* jre
  + bin – dll files
  + lib – files and folders, rt.jar

1. **Setting Environment Variables**

* Right click on This PC -> Properties
* Click on Advance System settings from the left panel
* Click on Environment Variables Button
* Under System Variable click on NEW ->
  1. Create JAVA\_HOME

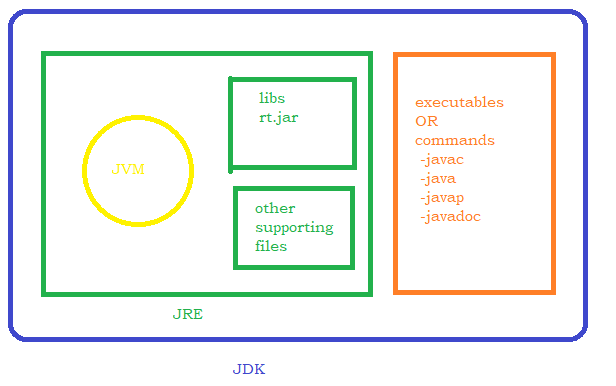
****

**2.** Select PATH Variable and click on edit,

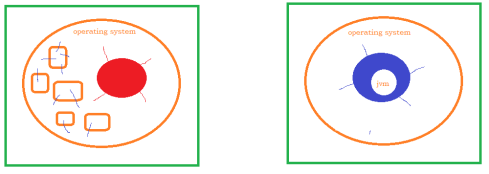


* Restart the command prompt and execute javac –version or java –version

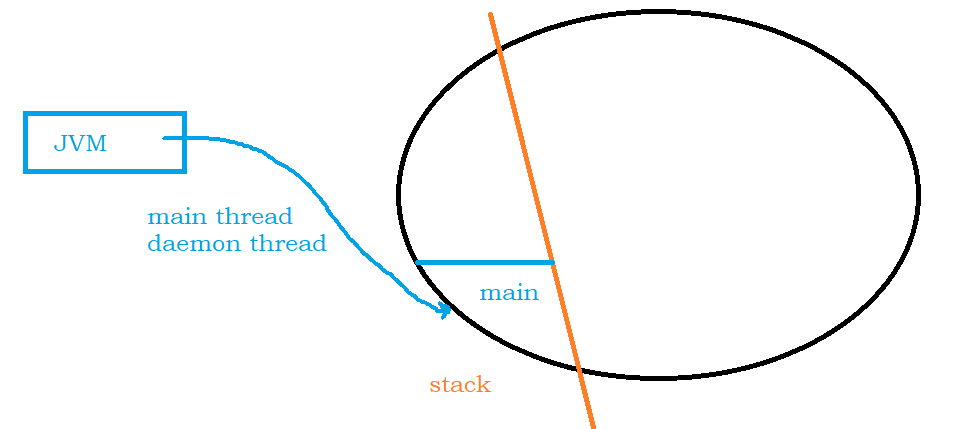
1. **JDK JRE JVM**

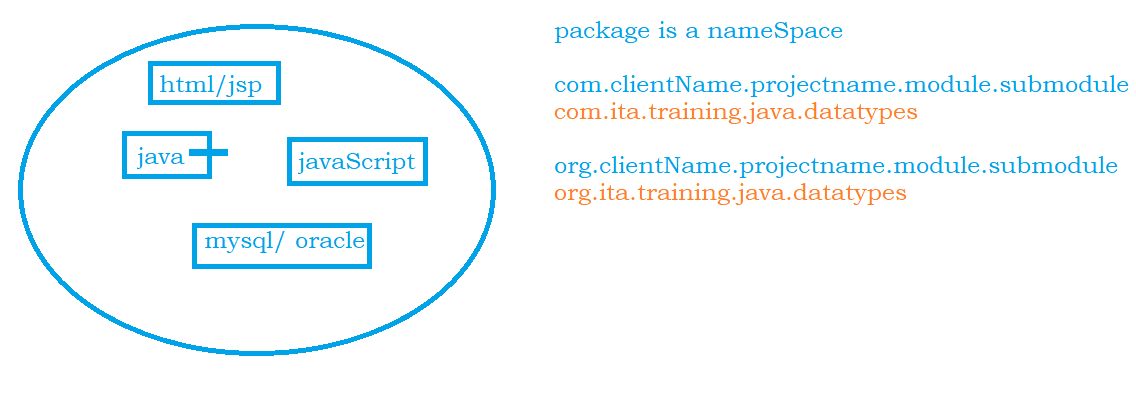
****

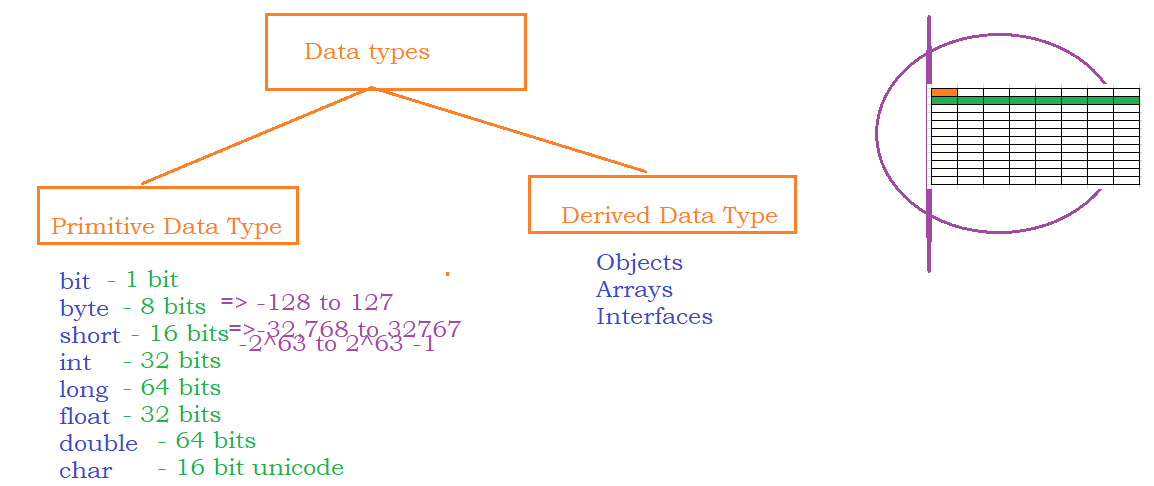
1. **Platform independent Nature**



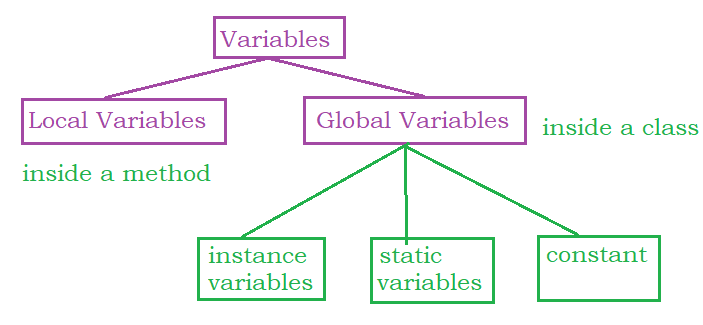
1. **Features**
   1. Open source free software
   2. Object Oriented
   3. Platform Independent
   4. Secured
   5. Simple
   6. Robust
   7. Architecture-Neutral
   8. Portable
   9. Multi-threaded
2. **Editor - Eclipse**
   1. Installation
      1. Go to google.com
      2. search for download eclipse
      3. [**https://www.eclipse.org/downloads/**](https://www.eclipse.org/downloads/)
      4. click on download packages - <https://www.eclipse.org/downloads/packages/>
      5. download eclipse for java developers
      6. unzip the downloaded zip file
      7. double click on eclipse.exe under eclipse folder
   2. Select Work Space -> Create your own folder and select it
   3. Create Project
   4. Create HelloWorld Java program
3. **HelloWorld Java Program**

****

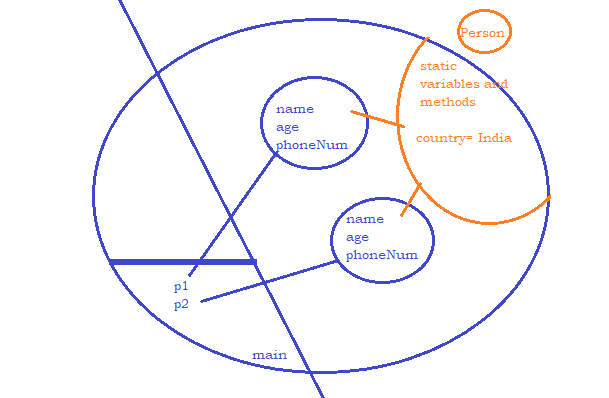
1. **Packages in java**
   1. ****
2. **Data types**

****

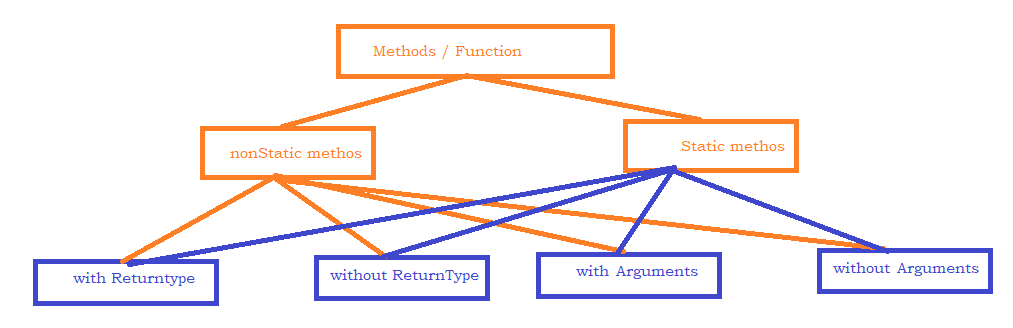
1. **Variables**

****

* **Global Variables**



1. **Methods**



[Access\_specifier] [Access\_modifier] return\_type name\_of\_theMethod( arguments list )

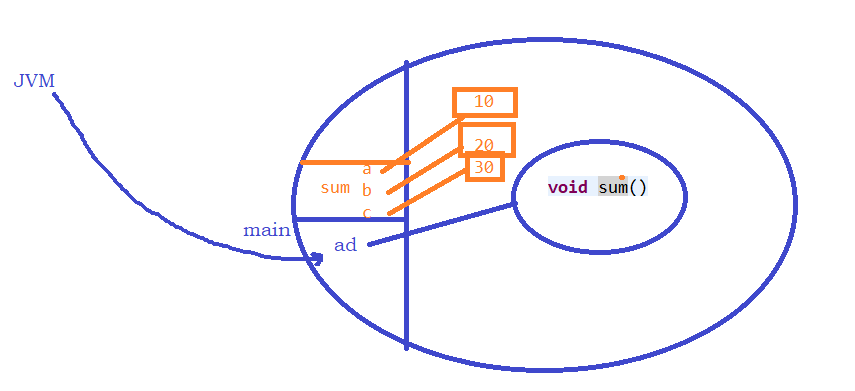
{

=====

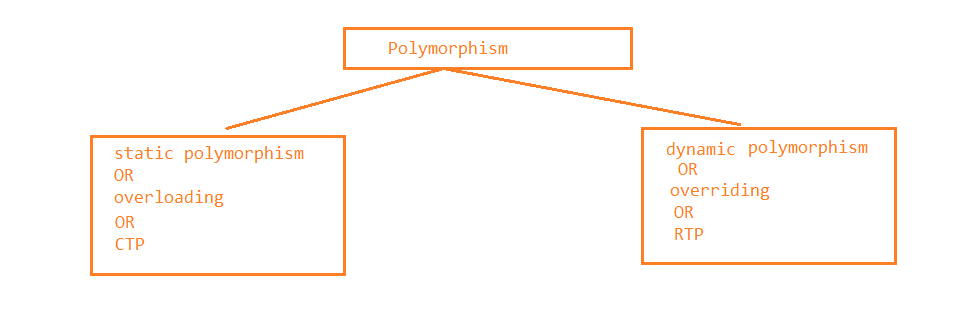
=====

}

**return\_type -> can be any data type (primitive / derived) OR void**

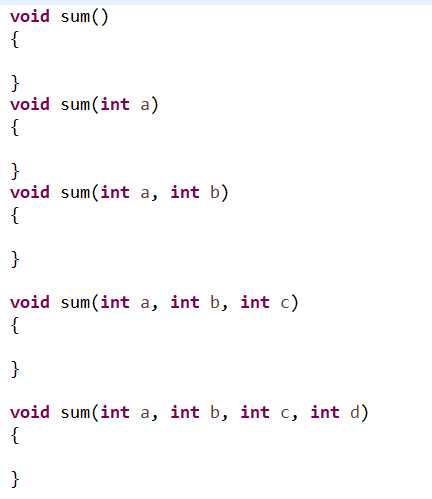
****

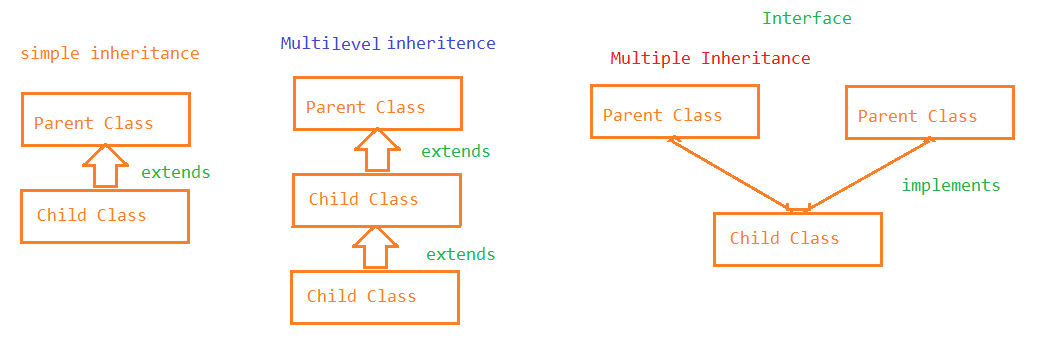
1. **Polymorphism**

****

1. **Type Casting –**

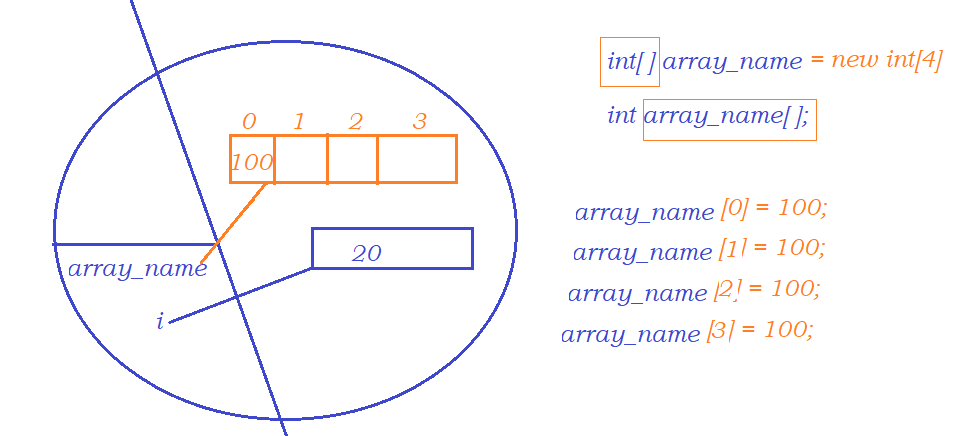
* **Auto type casting**
* **Explicit type casting**

1. **Variable number of arguments to a method**
   1. ****
2. **Inheritance**

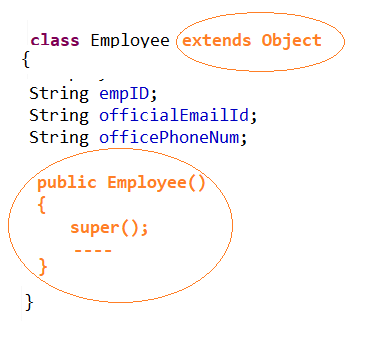
****

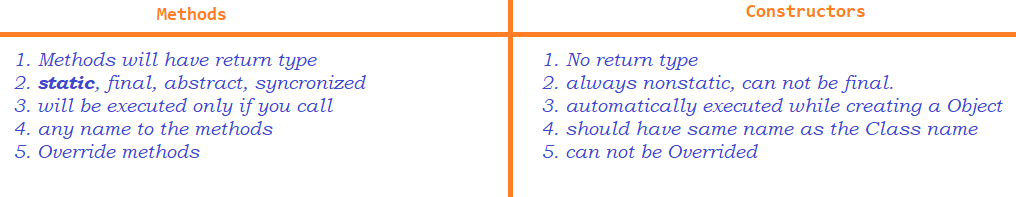
1. **Arrays**

Arrays are derived data types which can hold more than one value.



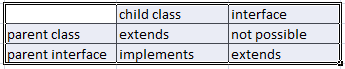
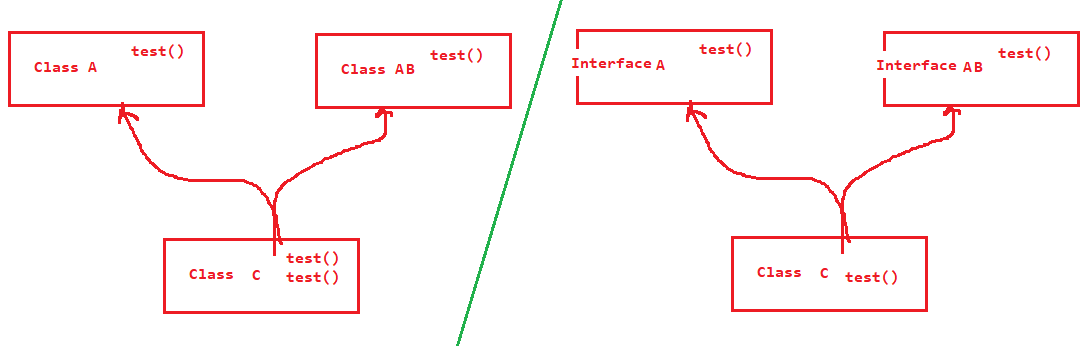
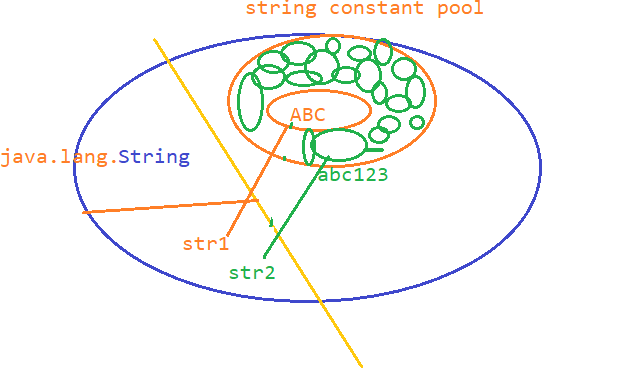
1. **Constructors**

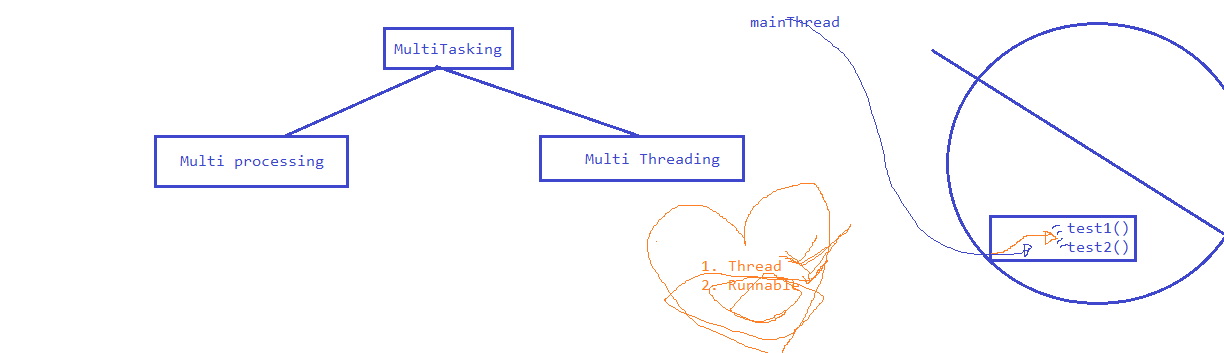
****

****

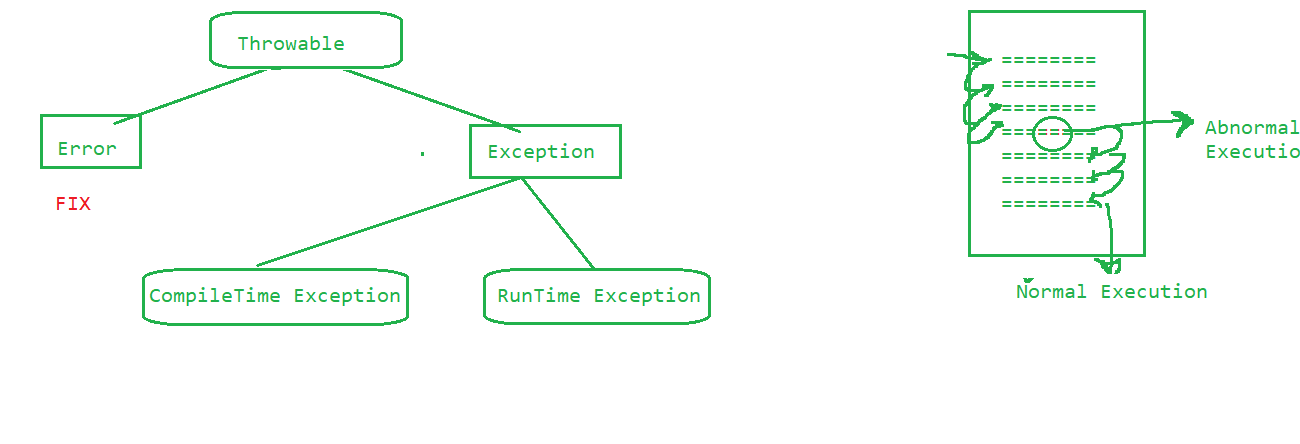
1. **Access specifiers** 
   1. private
   2. default (package)
   3. protected
   4. public
2. **Access Modifiers –** *Changes the behaviour of a member*
   1. static
   2. final
   3. abstract
   4. synchronized - *used to achieve thread safe in multihreaded programming*
3. **Abstract Classes in Java**
   1. if there are any unimplemented methods / abstract methods
   2. To restrict creating Object to the class

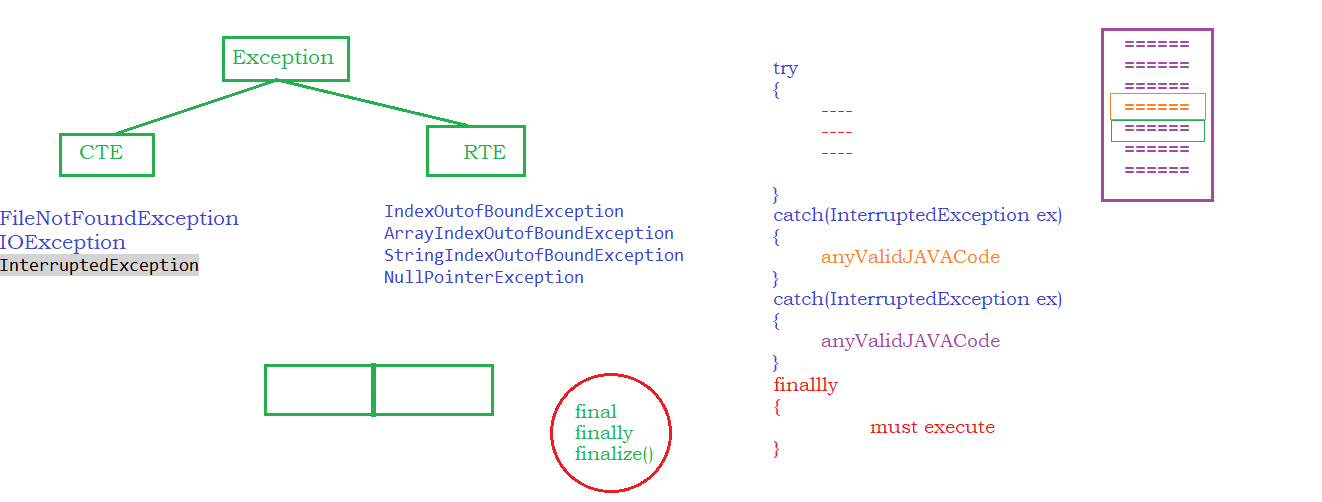
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | G-variables | methods | constructors | class | interfaces |
| private | yes | yes | yes | no | no |
| default | yes | yes | yes | yes | no |
| protected | yes | yes | yes | no | no |
| public | yes | yes | yes | yes | yes |
| static | yes | yes | no | no | no |
| final | yes | yes | no | yes | yes |
| abstract | no | yes | no | yes | yes |

1. **Interfaces** ->100% pure abstract class, to achieve multiple inheritance in java
2. **Inheritance**
   1. ****
   2. ****
3. **SIB / IIB**
4. **Arrays**
5. **looping and conditional statements**
6. **Enum**
7. **Strings**
   1. ****
8. **Threads**

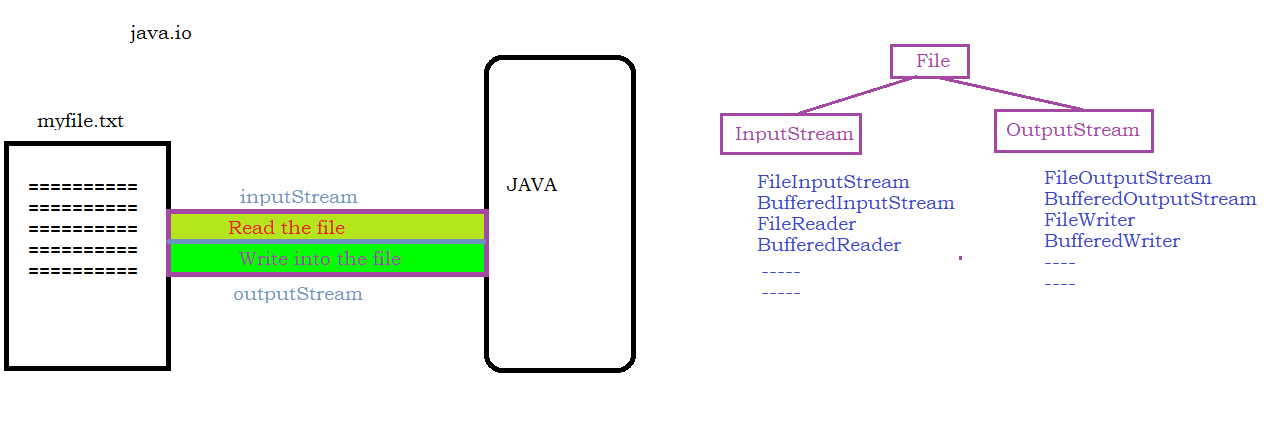
****

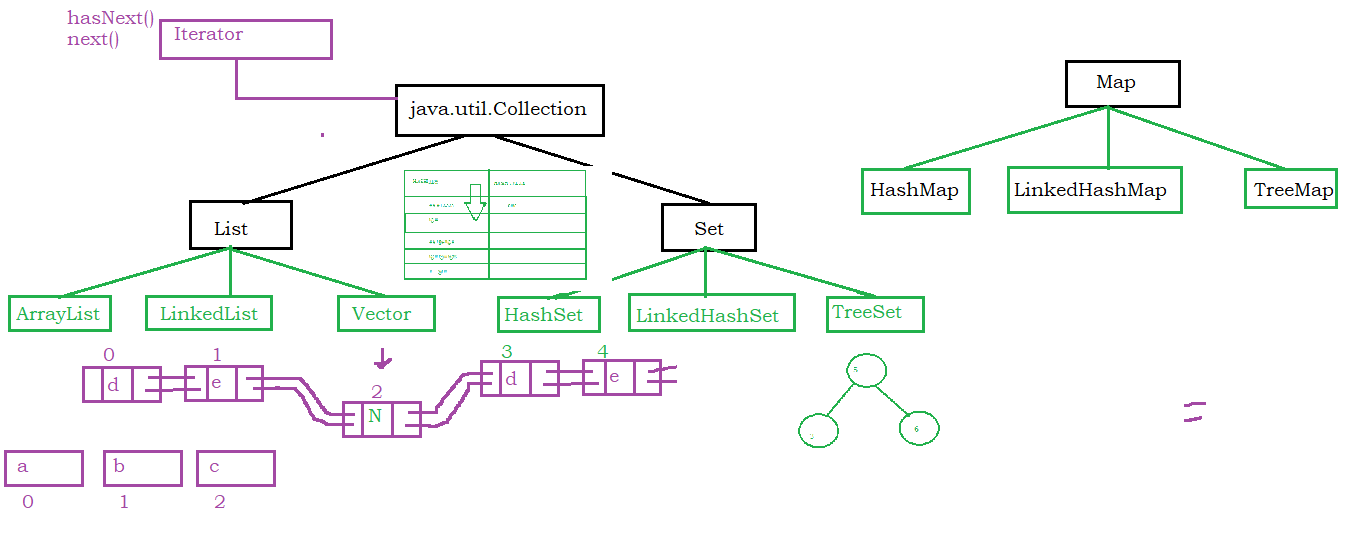
1. **Exception Handling**
   1. try
   2. catch
   3. throw
   4. throws
   5. finally



****

1. **File Handling**

****

1. **Collections**
2. ****
3. **Generics**
4. **TEST**