Java

Program – set of instructions

Machine level language – binary language – 0s and 1s

Compiler – Compiler converts program to machine understandable

format all at once and then execute.

int sum(int a ,int b){

return a+b;

}

Eg: c, c++, scala,smalltalk …

Interpreter- Interpreter converts each line of program to machine level

while it is executing.

Eg: Ruby,python etc..

Java is a platform and language.

Java follows WORA – write once run any where.

Features:

Simple

Platform Independent

Distributed

MultiThreaded

Robust

Secure

GarbageCollection- deallocation of memory when object its no longer being used- its taken care by jvm

Java is compiled and interpreted.

FileName.java – source code

compiled by javacompiler to bytecode- FileName.class

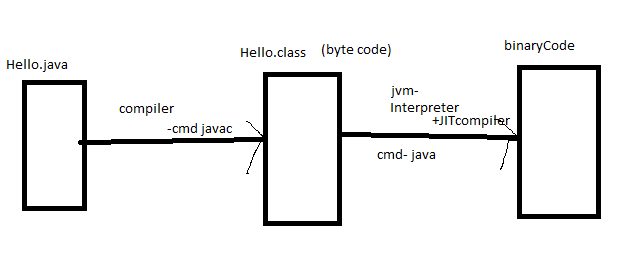
Interpreter converts to binary code and executes.

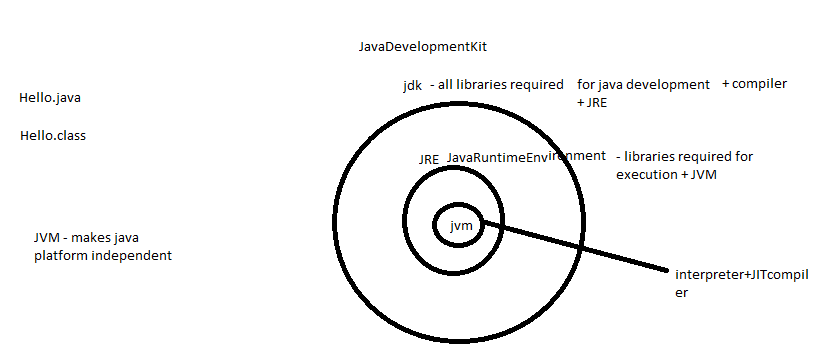
Java:

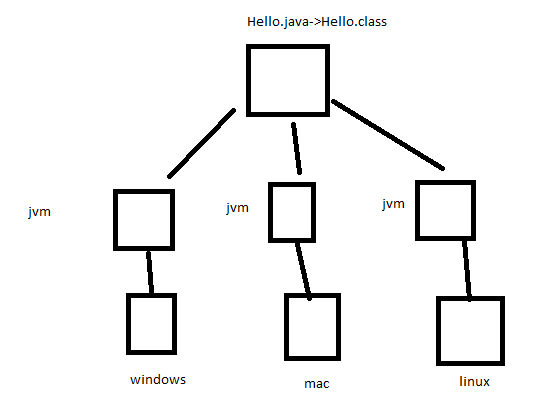
JDK – JavaDevelopmentKit

JRE- JavaRuntime environment

JVM-JavaVirtualMachine – jvm is system dependent but it makes JavaLanguage system independent.







jvm- interpreter + JIT(just in time) compiler

int sum(int a ,int b){

return a+b;

}

display all even numbers from 1-100

display(){

for(int i=1;i<=100;i++){

if(i%2==0){

print(I + “is even”);

}

}}

interpreter- 100\*1=100seconds + 3 seconds = 103seconds

JIT compiler – logic inside for loop is compiled to machine code in 1second , saved in memory and reused for all iterations.

1sec + 3 seconds – 4seconds

IDE : Integrated development environment which assists developer

to create, execute, debug and manage projects easily.

Java-Eclipse, Intellij-Ide’s-Jetbrains,Netbeans

Javascript- Webstorm , php storm

Android- AndroidStudio

IOS- xcode

Python- Pydev, pycharm

Open eclipse- create workspace-

Window-preferences- java- installed jre’s- point to jdk path instead of jre

Java is statically typed language

Javascript:

Var a=10;

Var a=”hello”;

Java:

Int a=10;

String a=”hello”;

Package : Package is a namespace/folder structure for organizing classes and interfaces

In a logical manner which helps projects easier to manage.

The first line in java program is package name.

Followed by import statements if any.

Followed by class declaration.

Class: Class is like a blue print from which objects are created. Class defines state and behavior

of object.

Student{

Properties-variables:state

Id

Name

Actions/functions- methods: behaviour

Training

Mocks

CodingTests

Interviews

}

Object: Object is the instance of class which defines state through variables and actions through methods.

Objects:

Student s1= new Student();

S1.name=”Ahad”;

S1.id=1;

Student s2= new Student();

S2.name=”Ahad”;

S2.id=1;

Student s3= new Student();

S3.name=”Ahad”;

S3.id=1;

**Keywords** : reserved by language for their implementations:

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/\_keywords.html

**AccessModifiers**:

Public

Private

Protected

Default/package

**DataTypes**:

Primitive

Objects

**Variables** :

AccessModifier datatype variablename;

Public String color;

**Java is case sensitive**

Datatype- What kind of data the variable will hold

Number

Characters

Setofcharacters- String

decimals

Boolean- true or false

Int – integer

String

**Method:**

It is a set of code which holds the actual logic and can be called at any point making

the code reusable.

greet(String name){

Print(“hello ”+ name);

}

Int add(int a,int b)

{

return a+b;

)

**Method signature(syntax) and method body(actual logic inside method):**

**Accessmodifier** **outputDataType/void** if u don’t return any **methodname**(**input arguments if any/or leave blank**){

…Logic…

**Return** data as per declared datatype in methodsignature

**No need to add return** statement if its void

}

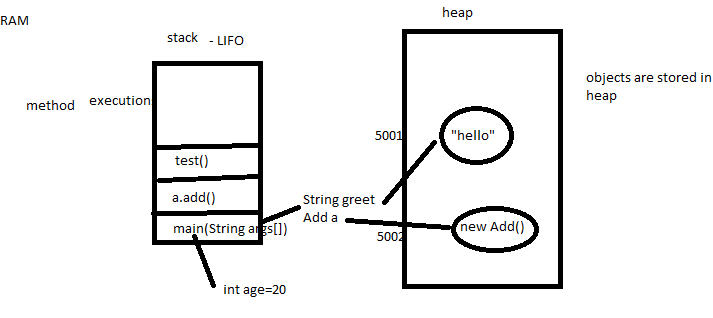
Always return statement is last line that wil be executed in a method.

**Compilation issues**- programmer deviates from language standards/ violates syntax rules.

Creating object in Java:

ClassName objname/reference variable = new ClassName();

Stack Heap:



Global or class level variables when not intialised with data will be given default values as per respective datatypes.

Object when they are not initialized – default value is null

For primitives it depends on datatype:

Eg- int-0, float-0.0

Create any class with properties and actions /call those methods/variables by creating object

Create a Calculator class with 4 methods- add, subtract,divide and multiply.

[training@whitebox-learning.com](mailto:training@whitebox-learning.com)

GarbageCollections- jvm deallocate the memory of objects in heap once it finds they are not being referenced from anywhere.

ctrl+shift+o- shortcut to import classes.

Constructor:

Constructor is similar to a method in java but it will not have return type, and it wil have same name as class name.

It is called by default whenever we create object.

It can take parameters which is a parameterized constructor.

Purpose of constructor is that it create an empty frame in memory

to reserve certain memory which is needed for object.

Uses of constructor:

Initialize any data required for the object because it wil be called initially When we create object.

A class can have more than one constructors with same class name

but different input parameters which is called constructor overloading.

2 ways to initialse class data:

Constructor Initialization

Setter Initialization

Access Modifiers: scope of class/method/variable is determined through access modifier.

Public – accessed anywhere within or outside the class

Private – accessed only within the class

Protected- accessed within the same package and also by subclasses

Outside the package.

protected variables can be accessed outside the package thru inheritance

default/package- accessed within the package

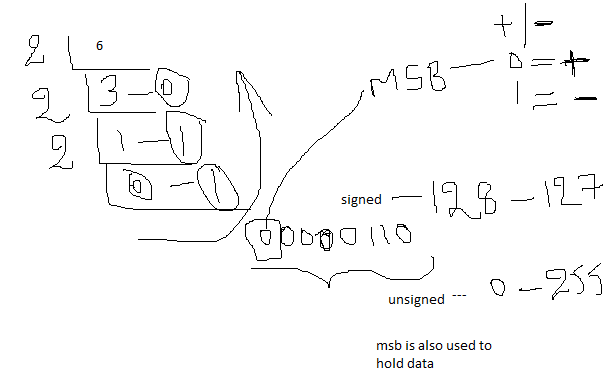
wider🡪narrow scopes:

public, protected ,default, private

all 4 access modifiers can be used w.r.to methods and variables

but for class we can use only public and default.

Datatypes:



Primitive: like literals, no need to use new keyword

Byte – integer with 8 bits => -128 to 127

Short- integer with 16 bits

Int- integer with 32 bits

Long- integer with 64 bits

Float - decimal with 32 bits

Double - decimal with 64 bits

Char – any keyboard character

Boolean- true or false

Other than primitive remaining all are objects in Java

byte->short->int->long->float->double

TypeCasting:

Explicit or downcasting – giving lower range values to higher range which we need to mention explicitly.

Implicit or upcasting – giving higher range to lower range which

Is done by default.

String – it is not primitive, it is class and we create objects for that class

Other than primitives remaining all are objects in java.

OOPS: Object oriented programming concepts

Encapsulation – hiding the implementation behind an interface/ binding the variables and methods together by making variables as private and giving public getters and setters.

Polymorphism

Inheritance

Abstraction