Day 1:

JAVA – Is a OOP language, platform Independent and also scalable.

Everything in Java is an Object.

Again we have classes where we write all our business logic etc..But objects are needed to work upon the classes.

To build web applications, Mobile applications etc..

Nowadays we have many built in frameworks which help us build the applications in a simple and robust way.

It is more structured so that it is maintenance easy.

Installation:

Download Java – JDK

Set environment variable

What are the OOP (Object Oriented Programming) concepts?

1. Abstraction
2. Encapsulation
3. Inheritance

Abstraction: Where I will not define things until required and only the necessary classes will do the implementation as per the logic needed. Hiding the implementation/details and not revealed to the outside world.

Abstract class policy{

Float calculatePolicyPremium();

}

Class JeevanBheem extends Policy{

Float calculatePolicyPremium(){

}

}

Class JeevanAnand extends Policy{

Float calculatePolicyPremium(){

}

}

Polymorphism:

One name many forms.

There are 2 types in polymorphism:

1. Method Overloading
2. Method Overriding
3. Method Overloading:
4. To satisfy method overloading, the methods should vary in the number of parameters or the datatype of the parameters.
5. The return value is not counted to meet the method overloading.

Class Example{

Int add(int a, int b){

}

Float add(int a, int b, int c){

}

Float add(float a, float b){

}

}

1. Method Overriding:

This comes up with your inheritance.

Class Parent{

Void display(){

// I am parent

}

}

Class Child extends Parent{

Void display(){ // overriding

// I am child

}

}

Child c = new child();

c.display(); // he will call child

suppose we commented the method (display()) in the child class

c.display() // parent will be called

Inheritance: // Reusability of the code, whatever Parent has flows down to the child class.

Class Parent{

Void buy(){

}

}

Class Child extends Parent{

// whatever the Parent buys in the above method is available to the child as well.

}

Class Car{

Model name; // Model is another class

float price;

String color;

String make;

}

Class BMW extends Car{

}

Class Audi extends Car{

}

In Java, everything Is an Object.

Simple example of Java class:

Class Student{

String name;

Int age;

String address;

String college;

//Subject subject;

Public static void main(String args[]){

}

}

Basic datatypes, Variables, Modifiers, Classes, objects

Primitives datatypes:

int, float, char, byte, long, double, String, boolean

int x = 5;

Reference data types:

These are user defined types.

Student, Employee, Banking -> Object types.

The class should contain a unique information. We need readability.

Class College{

String name;

Int dept\_count;

Department depts; -> another class // user defined type

}

Linkedlist :

Java also works by references.

Literals/Constants:

Some hard coded value for the variables.

X =5; str = “Manju”

Char c = ‘a’

pi = 3.14

Escape characters:

\n = newline operator

\t = tab

\b = backspace

Many keywords being supplied by Java:

New, return, void, String, int, instanceof, extends etc…

Variables: it’s a name given to the datatype to hold some data. Or it’s a named storage and can get manipulated.

Int x = 5; // x is a variable

String \_name24 = “CapG”; // is a valid variable

Different type of variables:

Local variables: This is the one which is defined inside a method or a block.

Instance variables: These are defined at the class level but outside the methods

Class/Static variables: These variables are declared using static keyword and these aren’t referenced through objects but rather through class name.

Access Modifiers:

There are 4 access modifiers:

Default: It is accessible within the package.

Public: visible to all!

Private: It is visible to only that class where it is defined.

Protected: visible to the package and also to the subclasses.

The other modifiers we have are:

static

final

abstract

Day 2:

Basic Operators, Conditional statements and Loops:

What is an operator? Which acts upon the variables and manipulates it!

Arithmetic operators, Relational operators, Bitwise operators, logical operators etc..

Int z = x+y;

X = 5

X++

++X

x- -

s.o.p(x++) 5

s.o.p(x) 6

s.o.p(++x) 6

for(int i=0;i<5;i++)

x = 5; Assignment operator

x+=5 ------🡪 x = x+5

x-=5 -------🡪 x = x-5

The below ones are the relational operators:

x==5; equality check

x!=5

x<=5

Bitwise operators

1100

0010

<<

>>

&&, !!, !

X&&Y

!(x&&Y)

Conditional / ternary operator: ? and :

(X==5) ? y=2 : y=3;

**Instanceof** operator:

Employee e = new Employee()

e is an instance of type Employee

e instanceof Student

Conditional statement:

If(profit > 10%){

s.o.p(“I am happy”)

} else{

s.o.p(“sorry!”)

}

If

Nested if

If – else

Switch statement

Its not compulsory for an if to have an else…

If(x==1){

}

If(x==2){

}

If(x==3){

} else{

Unknown number!

}

Int x = 5;

Switch (x){

case 1:

s.o.p(x);

case 2:

s.o.p(x);

case 5:

s.o.p(x);

default:

s.o.p(“default”);

}

Looping:

For()

While()

Do while()

When we need to iterate through multiple records for a similar sort of business logic.

Arrays: []

It is collection of homogeneous data.

Public static void main(String args[])

Scanner – args[0], args[1]

Int n[] = {1, 2, 3}

N is an array of type integers.

n.length = 3

n[0], n[1], n[2]

int a[];

a = new int[10]; // memory allocation happened…

a[] = {1, 2, 3, 4, 5}

a[0] = 1;

a[1] = 2;

for(int i=0;i<a.length;i++){

s.o.p(a[i])

}

i = a[2]

For(int i: a){

s.o.p(i);

}

While and do – while

Int x = 0;

Int a[] = {1,2,3}

While( x< a.length){

s.o.p(a[x])

x++;

}

The only difference between while and do-while is do-while will check the condition post execution of the while loop body.

do{

if(x == 4) {

return somevalue;

break;

}

x++;

} while(x<5);

Break and continue in the loops.

Inner:{

}