1. Closure in Java 8

Suppose we have execute method which has Process interface type argument

**private** **static** **void** execute(**int** x, Process p) {

System.***out***.println(" Summation is "+p.process(x));

}

**interface** Process{

**int** process(**int** i);

}

We can call the execute method like this

**int** b=3;

*execute*(2, **new** Process() {

@Override

**public** **int** process(**int** x) {

**return** x+b;

}

});

If this code is written in java 7 or any other old version, there will be a compilation error at line “int b =3” asking for convert b to final.

But from java 8 it will not throw any error, but if we try to change the value of b inside anonymous class call like

*execute*(2, **new** Process() {

@Override

**public** **int** process(**int** x) {

b =4;

**return** x+b;

}

});

At line “b=4” compiler will throw a compilation error say “Local variable b defined in an enclosing scope must be final or effectively final” that means the value of b is frozen or final insode the scope of the anonymous class.

1. this keyword in lambda:

Inspite lambda expression works alike an anonymous inner class, but if we use this keyword inside a lambda expression , it referes to the current context not the anonymous inner class context.

1. Use of lambda

**public** **interface** UserDefined1<T> {

**public** **void** apply(T t);

}

The interface can be generic or non generic

List<String> list = Arrays.*asList*("a","b","c");

UserDefined1<List<String>> obj1 = (l)->{for(String s : l) System.out.println(s)};

obj1.apply(list);

1. Use of colon: Method reference

Reference link : <https://netjs.blogspot.in/2015/06/method-reference-in-java-8.html>

Create a class ColonClassImpl and write a method of same signature like

apply. Return type does not matter. As here input argument of apply() is

generic, so any type can be allowed, but if apply(List<Integer> list) is there then it can be a

type mismatch compilation error

**class** ColonClassImpl{

**public** **void** method(List<String> list){

**for**(String s : list){

**if**(s.equalsIgnoreCase("a"))

System.***out***.println("A");

}

}

}

ColonClassImpl implObj = new ColonClassImpl();

UserDefined1< List<String>> obj2 = implObj :: method;