1.what is class resolution whether it is done in compile time or run time ?

A) Class resolution is basically getting references to the symbolic representation of different symbols present in the class.

2.(a)find the greatest of two or three numbers without using >,< symbol?

(b)without using && symbol

(A) class greatest

{

public static void main(String args[])

{

int a,b;

if(a/b==0)

system.out.println("b");

else

system.out.prientln("a");

}

}

3.Difference between elseif and switch case?

(A)(a)In else if, the control goes through the every else if statement until it finds true value of the statement or it comes to the end of the else if ladder.

In case of switch case, as per the value of the switch, the control jumps to the corresponding case.

(b)Switch case statement work on the basis of equality operator.

Whereas else if works on the basis of true false( zero/non-zero) basis.

(c)switch statement is considered to be less flexible than the else if ladder, because it allows only testing of a single expression against a list of discrete values.

4.Can we able to pass float,boolean,double values for switch?

(A)No,we are unable to pass float,boolean,double values for swich case.

5.why there is blank space after output?

(A)Because of system out println which moves the cursor to the next line.

6.What is the output of a& b, a| b ,~b, a^ b ?

(A)If we take a and b values as integers, compiler converts them into binary bits and perform bitwise operations and gives the output as integers.

Ex: a=9, b=7

a& b= 0001

a| b=1111

a^ b=1000

~b=1000

7.What are the differences between signed Int, Unsigned Int?

(A)**Signed Int:** It allows you to represent numbers in the positive and negative ranges.

**Unsigned Int:** It allows us only to represent numbers in the positive.

**Ex:** Unsigned values range 0 to 256, Signed values range -128 to 127

8. What are Unsigned right shift, Left shift , signed right shift ?

(A) **>> (Signed right shift)** In Java, the operator ‘>>’ is signed right shift operator. All integers are signed in Java, and it is fine to use >> for negative numbers. The operator ‘>>’ uses the sign bit (left most bit) to fill the trailing positions after shift. If the number is negative, then 1 is used as a filler and if the number is positive, then 0 is used as a filler. For example, if binary representation of number is **1**0….100, then right shifting it by 2 using >> will make it **11…1.**

**>>> (Unsigned right shift)** In Java, the operator ‘>>>’ is unsigned right shift operator. It always fills 0 irrespective of the sign of the number.

Unsigned right shift and left shit operator output will be same.