**Operator Precedence Chart in Java**

|  |  |
| --- | --- |
| **Operator** | **Precedence** |
| **Bracket** | ( ) |
| **Prefix increment ,decrement and unary** | **++ -- + - ~ !** |
| **Multiplicative** | **/ \* %** |
| **Additive** | **+ -** |
| **Postfix increment and decrement** | **++ --** |
| **Shift** | **<< >> >>>** |
| **Relational** | **< > <= >= instance of** |
| **Equality** | **== !=** |
| **Bitwise AND** | **&** |
| **Bitwise exclusive OR** | **^** |
| **Bitwise inclusive OR** | **|** |
| **Bitwise AND** | **&&** |
| **Bitwise OR** | **||** |
| **Ternary** | **? :** |
| **Assignment** | **= += -= \*= /= %= &= ^=** |

**Question 1**

public class P {

public static void main(String[] args) {

int a = 10, b = 5, c = 1, result;

result = a-++c-++b;

// result = a-(2)-(6)

// result = 10-2=8

// result = 8-6

// result = 2

System.out.println (result);

}

}

**Manual Output:** 2

**System Output:** 2

**Description:** As per the operator precedency in above example we have see the expression first assign the value of initialize it & start the execution of expression from the left to right like the result = 10 - ++1- ++5 then we take the increment value in the parenthesis with increment values like result = (10) – (2) – (6) & start execution from left to right & we get result = 8 – 6, result = 2.

**Question 2**

public class Second {

public static void main(String x1[]) {

int x = 0, y = 0 , z = 0 ;

x = (++x + y-- ) \* z++;

System.out.println("X is "+x);

}

}

**Manual Output:** x = 0

**System Output:** x = 0

**Description:** Here we see the x, y, z as ‘0’ initialized and execution is x = (++x+y--) \*z++; then put the values in expression x = (++0+0--)\*0++; after doing the increment & decrement the expression is x = (1+0--) \* 1; then x = (1- -) \* 1; then x = (0) \*1; & the final result is x = 0.

**Question 3**

class Numbers{

public static void main(String args[]){

int a=20, b=10;

if((a < b) && (b++ < 25)){

System.out.println("This is any language logic");

}

System.out.println(b);

}

}

**Manual Output:** 10

**System Output:** 10

**Description:** As we see above example value of a = 20 & b = 10 & the expression is ((a<b)&&(b++<25)) the first left side of expression is (a<b) means (20<10) here condition becomes false & the rule of && operator if left hand condition becomes false then right side expression was not check then if block statement not executed, so the result of above program is 10.

**Question 4**

class MyClass {

public static void main(String s[])

{ int i = 4;

int j = 21;

int k = ++i \* 7 + 2 - j--;

System.out.println("k = " + k+”\t J=”+j);

}

}

**Manual Output:** K= 16 J=20

**System Output:** K=16 J=20

**Description:** In above program the value of i = 4 & j = 21 an expression is int k = ++i \* 7 + 2 – j - - ; after the initialize values in expression k = ++4\*7+2-21--; let’s solve the expression with operator precedency & start the execution from left to right k = 5\*7+2-21--; then next step is k = 35+2-21--; because multiplication (\*) is more priority then k = 37-21--; after solve these k = 16; this is final result of k & after decrement the final value of j is 20.

**Question 5**

class IncDec {

public static void main(String s[])

{ int a = 1;

int b = 2;

int c;

int d;

c = ++b;

d = a++;

c++;

System.out.println ("a = " + a);

System.out.println ("b = " + b);

System.out.println ("c = " + c);

System.out.println ("d = " + d);

}

}

**Manual Output:**

a = 2

b = 3

c = 4

d = 1

**System Output:**

a = 2

b = 3

c = 4

d = 1

**Description:** Above program a is initialized 1 & b is 2. c & d is not initialized then next expression is c = ++b; here b is pre increment so the value of c is c = 3 & b = 3 then next expression is d = a++; here a is post increment so value of d = 1 & later value of a is 2 & next line c++ means previous value of c is 3 then now the increase by 1 & now value of c=4. Then print all values a = 2, b = 3, c = 4 & d = 1.

**Question 6**

public class Demo {

public static voidmain(String[] args)

{ int i, j, k, l = 0;

k = l++;

j = ++k;

i = j++;

System.*out*.println("I is "+i);

}

}

**Manual Output:** I is 1

**System Output:** I is 1

**Description:** Here above programs had declares four variables i, j, k, l=0 & next k = l++ here value of l is increment by 0 to 1 so value of k is also 0 then next step j = ++k; so the j = 1 because here k is pre increment and the next step i = j++; means value of j is post increment & current value of j is 1 so the value of i = 1 with output of l is 1.

**Question 7:**

public class Program7 {

public static void main(String[] args) {

int dailywage = 4;

int number\_of\_days = 5;

int salary;

salary = number\_of\_days++ \* --dailywage \* dailywage++ \* number\_of\_days--;

salary =- dailywage;

System.out.println(dailywage + " " + number\_of\_days + " " + salary);

}

}

Salary = 5++ \* --4 \* 4++ \* 5--;

Salary = 5++ \* 3 \* 3++ \* 5--;

Salary =

Salary = salary – dailywages;

**Manual Output**: 4 5 -4

**System Output:** 4 5 -4

**Description**: The above code we have the three variables dailaywage=4, number\_of\_days=5 and salary We see the expression of

salary=number\_of\_days++ \* --dailywage \*dailywage++ \* number\_of\_days-- in this expression top most priority operator is --dailywage and -- dailywage execute very first after executing --dailywage then these expression look like as ,

Salary = number\_of\_days++ \* --dailywage \* dailywage++ if we put the values in this expression salary = 5++ \* 3 \* 3++ so we have the two operator in expression ++ and \* but ++ in the form of post increment and post increment having less priority than \* so first multiplication get executed and result of multiplication stored in salary so salary= 45 after solve the multiplication ++ operator get executed number\_of\_days++ before increment number\_of\_days=5 so it will increase by 1 and number\_of\_days=6 and after that dailywage++ execute so initially dailywage = 4 but previous it is decrease by 1 with pre decrement operator so dailwage=3 at the time of multiplication but now dailywage++ execute so it will from 3 to 4 means now current value of dailywage=4 and after that number\_of\_days- - execute so number of days previous number\_of\_days=6 here again number\_of\_days decrease by 1 so it will again 5 and if we think about the statement salary = - dailywage so dailywage was 4 previous and when we initialize it in salary using negative sign then it will stored as -4 in salary and if we think about this statement System.***out***.println(dailywage + " " + number\_of\_days + " " + salary);

So we get the output 4 5 -4

**Question 8:**

class Program8 {

public static void main(String s[]){

int i = 34.0;

int j = 7;

int k = i % j;

System.out.println("k = " + k );

}

}

**Manual Output**: We will get compile time error.

**System Output:** We will get compile time error like that,

error: incompatible types: possible lossy conversion from double to int.

**Description**: Here we can store the double type of value in the int type of variable so that’s way we get this error.

**Question 9:**

class Program9 {

public static void main(String[] args){

int x = 42;

double y = 42.25;

System.out.println("x mod 10 = " + x % 10 );

System.out.println("y mod 10 = " + y % 10 );

}

}

**Manual Output:**

x mod 10 = 2

y mod 10 = 2.25

**System Output:**

x mod 10 = 2

y mod 10 = 2.25

**Description:** Above program we apply the % mod operator on double value & int value and we will get the result on that based on type.

**Question 10:**

Which of the following statements is true?

1. When a = 5 the value of a % 2 is same as a - 4

2. When a = 3 the value of a \* 3 \* 3 is greater than ( a + 10 ) \* 3

3. When a = 7 the value of a \* 7 \* 3 is greater than ( a \* a + 7 \* a + 3 )

**Manual Output**: 1 and 3

**System Output**: 1 and 3

**Description:**

- Statement 1: a = 5 -> a % 2 -> 5 % 2 = 1 and a - 4 = 5 - 4 = 1.

Both values are equal. So statement 1 is true.

- Statement 2: a = 3 -> a \* 3 \* 3 = 27 and (a + 10) \* 3 = ( 3 + 10 ) \* 3 = 13 \* 3 = 39.

But 27 is not greater than 39. So statement 2 is false.

- Statement 3: a = 7 -> a \* 7 \* 3 = 147 and ( a \* a + 7 \* a + 3 ) = ( 7 \* 7 + 7 \* 7 + 3 ) = ( 49 + 49 + 3 ) = 101.

So the 147 is greater than 101. So statement 3 is true.

**Question 11:**

public class Program11 {

public static void main(String[] args) {

System.*out*.println(10 \* 5 + 100 \* (25 \* 11) / (25 \* 10) \* 10 - 5 + 7 % 2);

int zx = (10 \* 5 + 100 \* (25 \* 11));

int yz = ((25 \* 10) \* 10 - 5 + 7 % 2);

System.*out*.println(zx / yz);

}

}

**Manual Output**:

1146

11

**System Output**:

1146

11

**Description:** In this code the expression given in the printing statement (10 \* 5 + 100 \* (25 \* 11) / (25 \* 10) \* 10 - 5 + 7 % 2); in that expression we have solve this by using precedency in that first we have solve based on precedency parenthesis then division, then multiplication then next modulo and finally we have do the addition & subtraction we get the result 1146. After this there are two variables are available with initialized the expression of it. First is int zx = (10 \* 5 + 100 \* (25 \* 11)) & next is int yz = ((25 \* 10) \* 10 - 5 + 7 % 2) in this also we have follow same rule and after getting result of it we have divide them as 27550/2496 and we get the ans is 11.

**Question 12:**

class Program12{

public static void main(String args[]) {

int var1 = 42;

int var2 = ~var1;

System.out.print(var1 + " " + var2);

}

}

**Manual Output:**  42 -43

**System Output:**  42 -43

**Description:** Here above program declare var1 & var2 and var1 initialized with 42 and print the it in below statement. Next var2 can initialize with negation of (tilt) operator after doing the decimal to binary conversion of 42 is 00101010 then apply tilt (~) operator and inverted the value is 11010101 & get the value of var2 is -43.

**Question 13:**

class Program13 {

public static void main(String args[]) {

int a = 3;

int b = 6;

int c = a | b;

int d = a & b;

System.*out*.println(c + "\t" + d);

}

}

**Manual Output:** 7 2

**System Output:** 7 2

**Description:** This code we get the binary value of 3 & 6 and apply the | and & operator in operator if both are 0 then answer is 0 otherwise 1 and in & operator if both bit are 1 then answer is 1 otherwise 0 and then calculate the we get the answer 7 & 2.

**Question 14:**

class Program14 {

public static void main(String args[])

{

byte x = 64;

int i;

byte y;

i = x << 2;

y = (byte) (x << 2);

System.*out*.print(i + "\t" + y);

}

}

**Manual Output:** 256 0

**System Output:** 256 0

**Description:** In above program we get the variable x, y, i and x can initialized 64 then take the i = x << 2; then we get binary 0100 0000 then we as per left shift operator rule we can shift the 0’s for left side as 0001 0000 0000 and we convert into in decimal we get the i = 256 and next line of expression y = (byte) (x << 2); as we see here the (x<<2) value get 256 but range of byte data type is +127 to -128 and then scale down value on these range then will get value of y is 0

**Question 15:**

class Program15 {

public static void main(String args[])

{

int x;

x = 10;

x = x >> 1;

System.*out*.println(x);

}

}

**Manual Output:** 5

**System Output:** 5

**Description:**  In above program we have x variable with value of 10 and the expression is x = x >> 1; then we take the 10 ‘s binary value then as per right shift rule shift 0’s by 1 for right side and after these binary to decimal value we get the value of x is 5.

**Question 16:**

class Program16 {

public static void main(String args[])

{

int a = 1;

int b = 2;

int c = 3;

a |= 4;

b >>= 1;

c <<= 1;

a ^= c;

System.*out*.println(a + " " + b + " " + c);

}

**Manual Output:** 3 1 6

**System Output:** 3 1 6

**Description:** In above code we have declare the variable a, b, c & initialize it 1, 2, 3 here first expression is a |= 4 as a = a | 4 here the we get binary value as 1 and 4 and do the | operation is 0001 | 0100 we get 0101 in decimal form of a = 5 in next expression b >>=1; as b = b >> 1 and here the we get binary value as 1 and 1 and do the | operation is 0001 | 0001 we get 0001 in decimal form **b = 1** next expression c <<= 1 ; as c = c << 1; here the we get binary value as 3 is 0011 and do the left shift operation after decimal operation we get as 0110 and its decimal is **c = 6** and there is one more expression is available a = a^c; means a = 5^6; in binary form is 0101^0110 = 0011 in decimal form is **a = 3.**

**Question 17:**

public class UnaryOperators {

public static void main(String[] args) {

int i = 5;

System.out.print(i++);

System.out.print(++i);

System.out.print(i--);

System.out.print(--i);

}

}

**Manual Output:** 5775

**System Output:** 5775

**Description**: In above code we have variable i = 5 them printing statement I++ but here first i is printing 5 then next increment by 6 then next line it is pre increment so first increment 6 to 7 and print as 7 then next post decrement so first print value as 7 then decrement it 6 and next statement pre decrement so value is first decrement 6 to 5 and print as 5 value and finally we get 5775.

**Question 18:**

public class ShiftOperators {

public static void main(String[] args) {

System.out.print(8<<2);

System.out.print(",");

System.out.print(8>>1);

}

}

**Manual Output:** 32,4

**System Output:** 32,4

**Description:** Here we do the binary value of 8 as 1000 then left shift of it by 2 then binary is 0010 0000 then its decimal is 32 and next print statement as ‘,’ then next same value with right shit 1000 as 1 like 0100 then its decimal is 4.

**Question 19:**

public class BitwiseNotOperator{

public static void main(String[] args) {

int i = 50;

System.out.print(~i);

System.out.print(",");

System.out.print(~--i);

System.out.print(",");

System.out.print(~++i);

}

}

**Manual Output:** -51,-50,-51

**System Output:** -51,-50,-51

**Description**: Here above code the we have taken the binary value of 50 as 0011 0010 and negation of that value 0 becomes 1 and 1 becomes 0 as like 1100 1101 then the rule of negation is first bit always reserved for sign bit so the take these decimal conversion and do the addition and we get the -51. Then next expression is ~---i; here value of I can be decrement by 1 as 50 to 49 then we do the 49 binary then next do the same of above expression & we get -50. Lastly expression is ~++I here previous value of I is 49 then increment by 1 as 50 do next same as first expression & we get -51.

**Question 20:**

public class RelationalOperators {

public static void main(String[] args) {

int a = 4;

int b = 5;

System.out.print(a>b);

System.out.print(",");

System.out.print(a<b);

}

}

**Manual Output:** false , true

**System Output:** false , true

**Description:** In above program value of a = 4 & b = 5 and first condition is a>b like 4>5 it becomes false and next condition is a<b like 4<5 it becomes true.

**Question 21:**

public class LogicalOperators {

public static void main(String[] args) {

int a = 5;

int b = 10;

boolean flag = a>b;

System.out.println(!flag);

}

}

**Manual Output**: true

**System Output:** true

**Description**: In above code the value of a & b is 5 & 10 respectively and condition is flag = a>b; so flag = 5 > 10; it is false but when we print the value then the these is !flag so in negation rule is false becomes true & true becomes false so the answer is true.

**Question 22:**

public class TernaryOperators {

public static void main(String[] args) {

int a = 10;

String result = a > 5 ? "Hello 1" : "Hello 2";

System.out.println(result);

}

}

**Manual Output:** Hello 1

**System Output:** Hello 1

**Description:** In these program the value of a = 10 and the condition is String result = a > 5 ? "Hello 1" : "Hello 2"; so 10 > 5 it is true then it return the Hello 1 as result.

**Question 23:**

class Ques

{

public static void main(String args[])

{

byte b = 12;

int y = b;

b = b + 10;

System.out.println(b);

}

}

**Manual Output:** In these program will get to compile time error as explicit casting is required in the line, b = b + 10.

**System Output:** to compile time error as explicit casting is required in the line, b = b + 10.

**Description**: When you compile the above program, a compile time error occurs in the line, b = b + 10 as explicit casting is required to assign an int value to a byte type. In the preceding program the byte type variable is declared and initialized to the value 12. Then the value of byte variable is assigned to the int variable, y as assigning byte type value to an int variable is possible. However the compilation error of loss of precision will occur while incrementing the byte type value with 10.

**Question 24:**

**Which of the following are valid declarations of the main () method?**

A. static main(String args[]){ }

B. public static String main(String args[]) {… }

C. public static void main(String args[]) {….}

D. final static void main(String args[]) {….}

**Manual Output:** The correct option is C.

**Description:** As per the java standard the correct implementation of main method is public static void main(String[] args) so the correct option is C.

**Question 25:**

**Ram as a developer was asked to create a program using switch…case within for loop. Ram created the following program:**

class Program25 {

public static void main(String args[])

{

int z=3;

for(int i=0; i<2;i++)

{

z++;

switch(z)

{

case 3:

System.out.print(z=z+1 +” “);

case 5:

System.out.print(z=z+2 + “ “);

break;

default :

System.out.print(z=z+8 + “ “);

case 6:

System.out.print( z=z+4 + “ “);

}

z--;

}

}

}

**Question 26:**

Imagine, you as a student provided with the following program during a class test

class Program26 {

public static void main(String args[]){

int z=6, k;

for(int i=0; i<2;i++) {

z++;

switch(z) {

case 3: System.out.print(z=z+1 + “ “);

case 5: System.out.print(z=z+2 + “ “);

break;

default : {

for (int x=10; x>3; x++) {

System.out.print(x=k+x + “ “);

}

}

case 6: System.out.print( z=z+4 + “ “);

}

z--;

}

}

}

What would be the output of the preceding program?

A. Program will display 8 10 as an output

B. Program will not compile successfully

C. Program runs infinity endless

D. Program will display 8 10 10as an output

**Question 27:**

**Ram as a student was provided with the following code snippet**

public void Ram(float c) {

switch (c) {

case 5:

case 7:

case 2:

default:

case 9.5:

}

}

After viewing the code snippet Ram was asked to notice the problems in preceding code snippet on the basis of the rules regarding switch…case statement. Following are the options from which Ram has to choose the correct answer.

A. There is no problem in the code snippet

B. Switch cannot evaluate float value

C. The default statement cannot be used between case statements

D. All cases must be in increasing order

**Manual Output:** Option B is correct.

**System Output:** error : Incompatible types double to float

**Description:** Here Option B is correct because switch case statement can evaluate to a char, byte, short, int, or enum. Therefore Option A is incorrect because float value is being used as a case, which is not allowed. Option C is incorrect because default can be used anywhere in switch…case block. Option D is incorrect because it is not necessary that cases must be in increasing order.

**Question 28:**

Sheela as a faculty given following options to her students and asked them to choose the correct options:

A. A switch statement can only evaluate to float and double values

B. A switch…case block must have break statements after every case

C. Switch case must be similar to switch expression type

D. A switch…case can be nested like nested if…else

**Output**: D is correct.

**Description**: Option D is correct because it is a fact.

Option A is incorrect because a switch…case cannot evaluate float and double.

Option B is incorrect because there is no need to have break statement after every case.

Option C is incorrect, for example, your case expression is of char type but you used 65 as case label then internally that 65 is recognized as char A. Therefore, case expression and case label can be varied but must take attention before using them. That’s way D option is correct.

**Question 29:**

Shyam during an interview was provided with following code and asked to review the program

public class Program29

{

public static void main(String args[])

{

int x=0, i=0;

for (int y=0; y>=i; ++y,i++) {

System.out.println(y);

System.out.println(i);

}

}

}

**After reviewing the code he was asked to predict the correct options among the following:**

A. Program will print 0 0 for first time

B. Program results in an endless loop

C. Program will not compile because declaration is not allowed inside the for loop

D. Program will successfully compile and print 0 0 on execution and then terminates

**Question 30:**

Rani during an interview was shown the following program:

class Program30 {

public static void main(String args[]) {

int x = 0;int y=9;

for ( ; x<y; ) { x++; y++;} // (a)

for (x; x==y; --x) continue; // (b)

for (x=0; x<5; ) { x++; } // (c)

for ( ; ; ) ; // (d)

}

}

What would be the output of the preceding program from the following options?

A. Program will successfully compile and executes but does not print any value

B. Program will successfully compile and becomes endless because of loop d

C. Program will not compile because loop b is syntactically incorrect

D. Program will not compile because loop a has only expression part but missing initialization and Increment/decrement part

**Question 31:**

Shyam was given the following code snippet during an interview and asked to choose all correct decisional and loop statements:

int y=9;

for ( ;true ; ) { break;} // 1

if(y==9) { break; } // 2

switch(y) {default: break;} // 3

do ( ){ // code } while(expression); // 4

while ( ) { //code } // 5

**Options:**

A. Statement 1 and 3 are correct

B. Statement 1, 2, and 3 are correct

C. Statement 1, 3, and 5 are correct

D. Statement 1,4, and 5 are correct

E. Statement 1, 2, and 4 are correct

**Question 32:**

Shyam was given the following program by his teacher

class Program32 {

public static void main(String args[])

{

int y=2;int i;

for (i=0; i <= 3; i++) {

if (i == 2) {

break;

}

else

{

y++;

}

}

System.out.println(i + ", " + y);

}

}

**What would be the output of the preceding code?**

A. Program will display 2 , 2 as an output

B. Program will display 2 , 3 as an output

C. Program will display 2 , 4 as an output

D. Program will display 1 , 2 as an output

**Question 33:**

Sheela after attending a lecture on for statement was shown the following for statements to choose the correct for statement:

A. for( int j=2; j\*j==4, j<4; j++)

B. for (int j=3; j/2==1; j++)

C. for (int j=3, long k=0; j>k; j++)

D. int k, j; for (j=3, k=2; k==j‐1; k++, j‐‐)

**Question 34:**

**Shyam works in a xyz company and he designed the following program:**

class Program34 {

public static void main (String args[]) {

int x=2; int y=6;

if( x!=y || (y\*=x)!=x) {

System.out.println(" Not equal");

}

else{

System.out.println(" Equal");

}

}

}

What happens when he compile and run the preceding program?

A. Program will display Equal

B. Program will display Not Equal

C. Program will not compile successfully because if statement is not correct

D. Program will compile but not executes

**Manual Output:** B is the correct answer**.**

**System Output:** Not equal

**Description:** Option B is correct because the first expression in if statement evaluates to true therefore Second expression is not checked and the statement in else block is displayed. Option A is incorrect because if statement evaluates to false. Options C and D are incorrect because the program compiles and executes successfully

**Question 35:**

**Rems and Sam while preparing for Java certification created the following program:**

public class Program35{

public static void main(String[] args)

{

char x = 'a';

switch(x)

{

case 66: System.out.println( "B" +” “);break;

case 72: System.out.println( "H"+ “ “);break;

case 97: System.out.println("a"+ “ +);

case 89: System.out.println( "Y" + “ “);break;

default: System.out.println( “default”);break;

}

}

}

**What would be the output of this program? Choose the correct option from the following options:**

A. Program will display a Y default

B. Program will display a Y

C. Program will not compile successfully because break cannot be used with default case.

D. Program will display A

**Manual Output:** B is the correct answer**.**

**System Output:** a Y

**Description:** Here Option B is correct because ASCII equivalent of small a is 97 and because break is not used so, statement following this case is also displayed. Option A is incorrect because break statement is used in case 89 (ASCII equivalent of Y).

Option C is incorrect because break can be used with default statement.

Option D is incorrect because ASCII equivalent of capital A is 65, which is not a case in this program.