OPERATORS MCQS

1. Which of these statements are incorrect?

- Assignment operators are more efficiently implemented by Java run-time system than their equivalent long forms
- Assignment operators run faster than their equivalent long forms
- Assignment operators can be used only with numeric and character data type
- **None of the mentioned**

Ans: **None of the mentioned**

truth value of **expression1**.

2) What will be the output of the following Java program? 1 point

```
class Modulus
{
    public static void main(String args[])
    {
        double a = 25.64;
        int  b = 25;
        a = a % 10;
        b = b % 10;
        System.out.println(a + " "  + b);
    }
}
```

- ◯ 5.640000000000001 5
- ◯ 5.640000000000001 5.0
- ◯ 5 5
- ◯ 5 5.640000000000001

Ans:**5.640000000000001 5**

- **a** is initially assigned the value 25.64, and then it is calculated as **a % 10**, which results in the remainder when 25.64 is divided by 10. The result is a floating-point number, and due to the precision of floating-point arithmetic, it becomes 5.640000000000001.
- **b** is initially assigned the value 25, and then it is calculated as **b % 10**, which results in the remainder when 25 is divided by 10. The result is an integer, which is 5.

So, the output of the program is "5.640000000000001 5".

3) What will be the output of the following Java program? 1 point

```
class increment
{
    public static void main(String args[])
    {
        double var1 = 1 + 5;
        double var2 = var1 / 4;
        int var3 = 1 + 5;
        int var4 = var3 / 4;
        System.out.print(var2 + " " + var4);

    }
}
```

- ◯ 1 1
- ◯ 0 1
- ◯ 1.5 1
- ◯ 1.5 1.0

Ans: C. **1.51**

Explanation:

- **var1** is assigned the value of the expression **1 + 5**, which is 6.
- **var2** is assigned the value of **var1 / 4**, which is 6 / 4, resulting in 1.5.
- **var3** is assigned the value of the expression **1 + 5**, which is 6.
- **var4** is assigned the value of **var3 / 4**, which is 6 / 4, resulting in 1 (as it's an integer division).

So, the output of the program is "1.5 1".

4) 1 point

What is the output of this program?

```
class bitwise_operator
{
    public static void main(String args[])
    {
        int a = 3;
        int b = 6;
    int c = a | b;
        int d = a & b;
        System.out.println(c + " "  + d);
    }
}
```

- ◯ 72
- ◯ 77
- ◯ 75
- ◯ 52

Ans: **Output c: 7 Output d: 2**

Explanation:

- It performs bitwise OR (|) and bitwise AND (**&**) operations on the integers **a** and **b**.
- **c = a | b** performs a bitwise OR operation, resulting in the binary **011 | 110** which is **111** in binary. Converting back to decimal, **c** becomes **7**.
- **d = a & b** performs a bitwise AND operation, resulting in the binary **011 & 110** which is **010** in binary. Converting back to decimal, **d** becomes **2**.
- Finally, "Output c: 7 output d: 2"

What is the output of this program?

```
class bool_operator
{
    public static void main(String args[])
    {
        boolean a = true;
        boolean b = !true;
        boolean c = a | b;
    boolean d = a & b;
        boolean e = d ? b : c;
        System.out.println(d + " " + e);
    }
}
```

- ◯ false false
- ◯ true true
- ◯ true false
- ◯ false true

Ans: False True Explanation:

- **a** is assigned the value **true**.
- **b** is assigned the negation of **true**, which is **false**.
- **c** is assigned the result of the bitwise OR (|) operation between **a** and **b**, which is **true | false** resulting in **true**.
- **d** is assigned the result of the bitwise AND (**&**) operation between **a** and **b**, which is **true & false** resulting in **false**.
- **e** is assigned the value of **d ? b : c**, since **d** is **false**, **e** becomes **c**, which is **true**.

Therefore, the output is "false true".

6) Which of these is returned by greater than, <, and equal to, ==, operator?

- Integers
- **Floating - point numbers**
- Boolean
- None of the mentioned

Ans: Floating - point numbers

7). Which of these have highest precedence?

- ()
- ++
- *
- >>

Ans:**() (parentheses)**

In Java, parentheses have the highest precedence. They are used to group expressions and override the default order of evaluation.

8). What is the value stored in x in following lines of code?

int x, y, z; x = 0; y = 1; x = y = z = 8;

- 0
- 1
- 9
- **8**

Ans: 8

The entire expression **x = y = z = 8** is evaluated from left to right.

o **y = 8**: First, **y** is assigned the value **8**. o **z = 8**: Now, **z** is

assigned the value **8**.

o **x = 8**: Finally, the result of the previous operation (which is **8**) is assigned

to **x**.

9) Which is the Logical operator in Java that works with a Single Operand?

- Logical AND
- Logical OR
- Logical Exclusive OR
- **Logical NOT**

Ans: Logical NOT

The Logical NOT operator, represented by the ! symbol, is the only logical operator in Java that works with a single operand. It reverses the value of the operand:

• If the operand is true, it becomes false. • If the operand is false, it becomes true.

10. What should be expression1 evaluate to in using ternary operator as in this line? expression1 ? expression2 : expression3

- Integer
- Floating " “ point numbers
- **Boolean**
- None of the mentioned

Ans: **Boolean**

The ternary operator in Java requires the conditional expression (**expression1**) to evaluate to a boolean value. It decides whether to execute **expression2** or **expression3** based on the