

## Selection 2

Operation	Example
<code>if</code>	<code>if (x &gt; 0) { sout("Positive"); }</code>
<code>if ... else</code>	<code>if (x &gt; 0) { sout("Positive"); } else { sout("Non-positive"); }</code>
<code>if ... elseif ... else</code>	<code>if (x &gt; 0) { sout("Positive"); } else if (x &lt; 0) { sout("Negative"); } else { sout("Zero"); }</code>
<code>switch</code>	<code>switch (x) { case 1: sout("One"); break; case 2: sout("Two"); break; default: sout("Other"); }</code>
Nested <code>if</code> statement	<code>if (x &gt; 0) { if (x % 2 == 0) { sout("Positive and Even"); } }</code>
Conditional Operators	<code>boolean isTrue = (x &gt; 0) &amp;&amp; (x &lt; 10);</code>

Operation	Description	Example	Output Example
<code>+</code> (Binary addition)	Binary arithmetic addition.	<code>int a = 5 + 3;</code>	a equals 8
<code>-</code> (Binary subtraction)	Binary arithmetic subtraction.	<code>int b = 8 - 4;</code>	b equals 4
<code>*</code> (Multiplication)	Binary multiplication.	<code>int c = 6 * 4;</code>	c equals 24
<code>/</code> (Division)	Binary division.	<code>int d = 20 / 5;</code>	d equals 4
<code>%</code> (Modulus)	Binary modulus (remainder of division).	<code>int e = 17 % 5;</code>	e equals 2
<code>+</code> (Unary plus)	Unary plus.	<code>int f = +5;</code>	f equals 5
<code>-</code> (Unary minus)	Unary minus.	<code>int g = -7;</code>	g equals -7
<code>++var</code> (Pre-increment)	Pre-increment.	<code>int h = 5; ++h;</code>	h becomes 6
<code>--var</code> (Pre-decrement)	Pre-decrement.	<code>int i = 10; --i;</code>	i becomes 9
<code>var++</code> (Post-increment)	Post-increment.	<code>int j = 5; j++;</code>	j becomes 6
<code>var--</code> (Post-decrement)	Post-decrement.	<code>int k = 10; k--;</code>	k becomes 9
<code>!</code> (Not)	Logical NOT.	<code>boolean l = !flag;</code>	l becomes false
<code>&lt;</code> (Less than)	Less than comparison.	<code>boolean m = (5 &lt; 10);</code>	m becomes true
<code>&gt;</code> (Greater than)	Greater than comparison.	<code>boolean n = (10 &gt; 5);</code>	n becomes true
<code>!=</code> (Not equal to)	Not equal to comparison.	<code>boolean o = (3 != 5);</code>	o becomes true
<code>==</code> (Equal to)	Equal to comparison.	<code>boolean p = (5 == 5);</code>	p becomes true
<code>&lt;=</code> (Less than or equal to)	Less than or equal to comparison.	<code>boolean q = (10 &lt;= 15);</code>	q becomes true
<code>&gt;=</code> (Greater than or equal to)	Greater than or equal to comparison.	<code>boolean r = (20 &gt;= 10);</code>	r becomes true
<code>&amp;&amp;</code> (Logical AND)	Conditional AND.	<code>boolean s = (true &amp;&amp; false);</code>	s becomes false
<code>  </code> (Logical OR)	Conditional OR.	<code>boolean t = (true    false);</code>	t becomes true
<code>^</code> (Logical XOR)	Conditional XOR (exclusive OR).	<code>boolean u = (true ^ false);</code>	u becomes true
<code>=</code> (Assignment)	Assignment.	<code>int w = 5;</code>	w becomes 5

Operation	Description	Example	Output Example
<b>+=</b> (Add and assign)	Add and assign.	<code>int x = 5; x += 3;</code>	x becomes 8
<b>-=</b> (Subtract and assign)	Subtract and assign.	<code>int y = 10; y -= 4;</code>	y becomes 6
<b>*=</b> (Multiply and assign)	Multiply and assign.	<code>int z = 20; z *= 2;</code>	z becomes 40
<b>/=</b> (Divide and assign)	Divide and assign.	<code>int m = 15; m /= 3;</code>	m becomes 5
<b>%=</b> (Modulus and assign)	Modulus and assign.	<code>int n = 12; n %= 5;</code>	n becomes 2

1. Fill the blanks so the code prints Path A.

```
class punchcard {  
    public static void main(String[] args) {  
        int x = 4;  
        ----- (x) {  
            ----- 4:  
                System.out.println("Path A");  
            -----;  
        default:  
            System.out.println("Path B");  
            break;  
        }  
    }  
}
```

- A) switch, case, break
  - B) if, else, continue
  - C) if, else, break
  - D) switch, case, continue
- 

2. Given the following code snippet, what will be the output?

```
class punchcard {  
    public static void main(String[] args) {  
        int y = 4;  
        switch (y) {  
            default:  
                System.out.println("Default");  
            case 1:  
                System.out.println("One");  
                break;  
            case 2:  
                System.out.println("Two");  
            case 3:  
                System.out.println("Three");  
                break;  
        }  
    }  
}
```

- A) Default
- B) One
- C) Default \n One

- D) Default \n One \n Two \n Three
  - E) Default \n Two \n Three
- 

3. What is the output of the following Java code?

```
class punchcard {  
    public static void main(String[] args) {  
        int z = 2;  
        switch (z) {  
            case 1:  
                System.out.println("Case 1");  
                break;  
            case 2:  
            case 3:  
                System.out.println("Case 2 or 3");  
                break;  
            default:  
                System.out.println("Default");  
        }  
    }  
}
```

- A) Case 1
  - B) Case 2 or 3
  - C) Default
  - D) Case 2 or 3 \n Default
  - E) No output
  - F) Error
- 

4. What is the output of the following Java code?

```
class punchcard {  
    public static void main(String[] args) {  
        int input = 7;  
        switch (input % 5) {  
            case 1:  
                System.out.println("One");  
                break;  
            case 2:  
                System.out.println("Two");  
                break;  
            case 3:  
                System.out.println("Three");  
                break;  
            case 4:  
                System.out.println("Four");  
                break;  
            default:  
                System.out.println("Default");  
        }  
    }  
}
```

```

        System.out.println("Three");
        break;
    default:
        System.out.println("Default");
    }
}
}

```

- A) One
- B) Two
- C) Three
- D) Default

- 
- Question 5 to 7 are extra. You can skip them if you want.
  - These switch features require Java 14 or higher.

5. What is the output of the following Java code?

```

class punchcard {
    public static void main(String[] args) {
        int dayofweek = 6;
        switch (dayofweek) {
            case 1, 2, 3, 4, 5:
                System.out.println("wait, we have class today?!!");
                break;
            case 6, 7:
                System.out.println("going to a partttttttttttty");
                break;
            default:
                System.out.println("not a valid day!");
        }
    }
}

```

- A) wait, we have class today?!!
- B) going to a partttttttttttty
- C) Not a valid day!
- D) Error

---

6. What is the output of the following Java code?

```

[class](class) punchcard {
    public static void main(String[] args) {

```

```

int dayOfWeek = 6;
switch (dayOfWeek) {
    case 1, 2, 3, 4, 5 -> System.out.println("wait, we have class today?!");
    case 6, 7 -> System.out.println("going to a partttttttttttty");
    default -> System.out.println("Not a valid day!");
}
}
}

```

- A) wait, we have class today?!!
- B) going to a partttttttttttty
- C) Not a valid day!
- D) Error

---

7. What is the output of the following Java code?

```

class punchcard {
    public static void main(String[] args) {
        int dayOfWeek = 7;
        int day = 23, month = 5, year = 2021;

        String date = switch (dayOfWeek) {
            case 1 -> "Monday";
            case 2 -> "Tuesday";
            case 3 -> "Wednesday";
            case 4 -> "Thursday";
            case 5 -> "Friday";
            case 6 -> "Saturday";
            case 7 -> "Sunday";
            default -> "Invalid day of week!";
        }
        + ", " + day + ". "
        + switch (month) {
            case 1 -> "January";
            case 2 -> "February";
            case 3 -> "March";
            case 4 -> "April";
            case 5 -> "May";
            case 6 -> "June";
            case 7 -> "July";
            case 8 -> "August";
            case 9 -> "September";
            case 10 -> "October";
        }
    }
}

```

```

        case 11 -> "November";
        case 12 -> "December";
        default -> "Invalid month!";
    }
    + " " + year;
    System.out.println(date + "\n");
}
}

```

- A) Sunday, 23. May 2021
- B) Invalid day of week!, 23. Invalid month! 2021
- C) Invalid day of week!, 23. May 2021
- D) Thursday, 23. April 2021
- E) Error

---

8. You are creating a program to simulate a color-matching game where each color corresponds to a specific number. Your task is to write a code that takes a user input of type string (`color`). The input color is matched with a numeric code based on the following color chart:

The color codes are as follows: - Red - 101 - Blue - 202 - Green - 303 - Yellow - 404 - Orange - 505

Then print the corresponding numeric code to the console.

Example function signature:

```

public class Main {
    public static void main(String[] args) {
        // make a scanner object

        // get the color from the user

        // use if-else or switch-case to print the corresponding numeric code
    }
}

```

#### Constraints:

- The input `color` will always be a valid string among: “Red”, “Blue”, “Green”, “Yellow”, “Orange”.

#### Example:

This question challenges candidates to use a **switch-case** statement to match the input color string with its corresponding numeric code efficiently and creatively. The goal is to implement a compact and elegant `getColorCode` function using the switch-case construct.

Answers:

1. A
2. C
3. B
4. B
5. B
6. B
7. A
8. a possible solution:

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String color = scanner.nextLine();
        switch (color) {
            case "Red":
                System.out.println("101");
                break;
            case "Blue":
                System.out.println("202");
                break;
            case "Green":
                System.out.println("303");
                break;
            case "Yellow":
                System.out.println("404");
                break;
            case "Orange":
                System.out.println("505");
                break;
            default:
                System.out.println("Invalid color");
        }
    }
}
```