

## WEEK END TASK JAVA

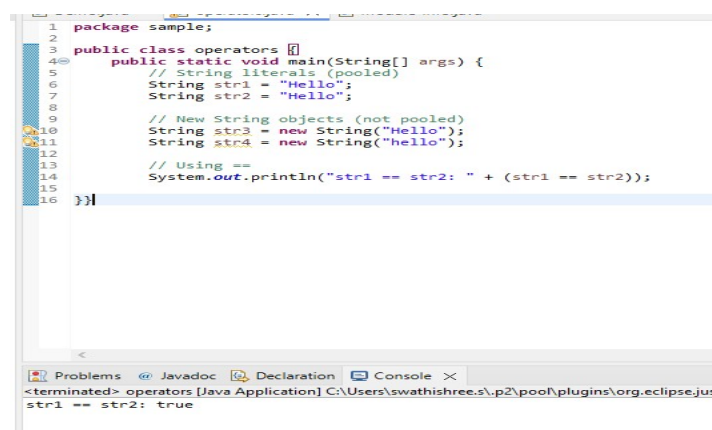
### CASE: 1

### QUESTION: 1

```
public class StringComparisonExample {  
    public static void main(String[] args) {  
        // String literals (pooled)  
        String str1 = "Hello";  
        String str2 = "Hello";  
  
        // New String objects (not pooled)  
        String str3 = new String("Hello");  
        String str4 = new String("hello");  
  
        // Using ==  
        System.out.println("str1 == str2: " + (str1 == str2)); // 1. (same memory reference)  
        what's the result?  
        System.out.println("str1 == str3: " + (str1 == str3)); //2. (different memory references)  
        what's the result?  
  
        // Using equals()  
        System.out.println("str1.equals(str3): " + str1.equals(str3)); //3. (same content) what's  
        the result?  
        System.out.println("str1.equals(str4): " + str1.equals(str4)); //4. (case-sensitive) what's  
        the result?  
  
        // Using equalsIgnoreCase()  
        System.out.println("str1.equalsIgnoreCase(str4): " + str1.equalsIgnoreCase(str4)); //5.  
        (case-insensitive) what's the result?  
    }  
}
```

### ANSWER:

- 1) str1 == str2:  
Result: True



The screenshot shows an Eclipse IDE window with a Java file named 'operators.java'. The code in the file is as follows:

```
1 package sample;  
2  
3 public class operators {  
4     public static void main(String[] args) {  
5         // String literals (pooled)  
6         String str1 = "Hello";  
7         String str2 = "Hello";  
8  
9         // New String objects (not pooled)  
10        String str3 = new String("Hello");  
11        String str4 = new String("hello");  
12  
13        // Using ==  
14        System.out.println("str1 == str2: " + (str1 == str2));  
15  
16    }  
}
```

The console window at the bottom shows the output of the program:

```
<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.jdt  
str1 == str2: true
```

2) str1 == str3:

Result: False

```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5         // String literals (pooled)
6         String str1 = "Hello";
7         String str2 = "Hello";
8
9         // New String objects (not pooled)
10        String str3 = new String("Hello");
11        String str4 = new String("hello");
12
13        // Using ==
14        System.out.println("str1 == str3: " + (str1 == str3));
15
16    }}
```

<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre

str1 == str3: false

3) str1.equals(str3):

Result: True

```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5         // String literals (pooled)
6         String str1 = "Hello";
7         String str2 = "Hello";
8
9         // New String objects (not pooled)
10        String str3 = new String("Hello");
11        String str4 = new String("hello");
12
13        // Using equals()
14        System.out.println("str1.equals(str3): " + str1.equals(str3));
15
16    }}
```

<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre

str1.equals(str3): true

4) str1.equals(str4):

Result: False

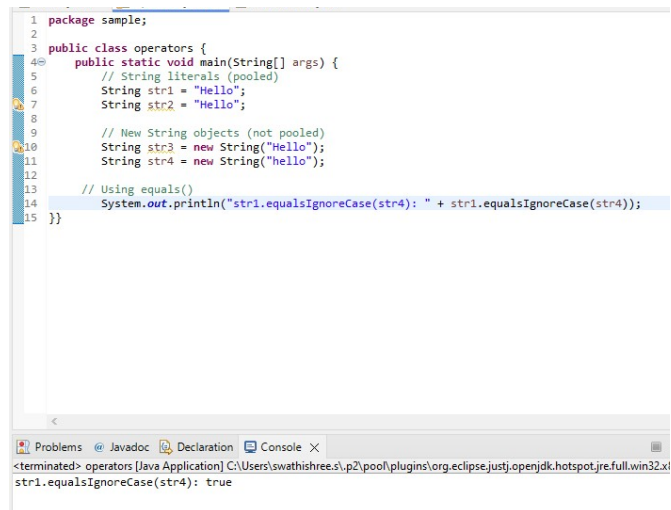
```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5         // String literals (pooled)
6         String str1 = "Hello";
7         String str2 = "Hello";
8
9         // New String objects (not pooled)
10        String str3 = new String("Hello");
11        String str4 = new String("hello");
12
13        // Using equals()
14        System.out.println("str1.equals(str4): " + str1.equals(str4));
15    }}
```

<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openjdk.h

str1.equals(str4): false

5) str1.equalsIgnoreCase(str4):

Result: True



```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5         // String literals (pooled)
6         String str1 = "Hello";
7         String str2 = "Hello";
8
9         // New String objects (not pooled)
10        String str3 = new String("Hello");
11        String str4 = new String("hello");
12
13        // Using equals()
14        System.out.println("str1.equalsIgnoreCase(str4): " + str1.equalsIgnoreCase(str4));
15    }
16 }
```

Problems Javadoc Declaration Console X

<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64.jdk\bin\java.exe -cp C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64.jdk\bin\java.exe operators

## CASE:2

### QUESTION:2

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

//Mention what's the result in 1, 2, 3,4 and 5

// Primitive int

int int1 = 100;

int int2 = 100;

// Integer objects

Integer intObj1 = 100;

Integer intObj2 = 100;

Integer intObj3 = new Integer(100);

Integer intObj4 = new Integer(200);

// Using == with primitive int

System.out.println("int1 == int2: " + (int1 == int2)); // 1. (compares values)

// Using == with Integer objects (within -128 to 127 range)

System.out.println("intObj1 == intObj2: " + (intObj1 == intObj2)); // 2. (cached objects)

// Using == with Integer objects (new instance)

System.out.println("intObj1 == intObj3: " + (intObj1 == intObj3)); // 3. (different instances)

// Using equals() with Integer objects

System.out.println("intObj1.equals(intObj3): " + intObj1.equals(intObj3)); // 4. (same content)

System.out.println("intObj1.equals(intObj4): " + intObj1.equals(intObj4)); // 5. (different content)

```
}  
}
```

## ANSWER:

1) `int1 == int2:`

Result: True

```
1 package sample;  
2  
3 public class operators {  
4     public static void main(String[] args) {  
5  
6         //Mention what's the result in 1, 2, 3,4 and 5  
7         // Primitive int  
8         int int1 = 100;  
9         int int2 = 100;  
10  
11        // Integer objects  
12        Integer intObj1 = 100;  
13        Integer intObj2 = 100;  
14        Integer intObj3 = new Integer(100);  
15        Integer intObj4 = new Integer(200);  
16  
17        // Using == with primitive int  
18        System.out.println("int1 == int2: " + (int1 == int2)); // 1. (compares values)  
19    }  
}
```

Problems Javadoc Declaration Console X  
<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre1  
int1 == int2: true

2) `intObj1 == intObj2:`

Result: True

```
1 package sample;  
2  
3 public class operators {  
4     public static void main(String[] args) {  
5  
6         //Mention what's the result in 1, 2, 3,4 and 5  
7         // Primitive int  
8         int int1 = 100;  
9         int int2 = 100;  
10  
11        // Integer objects  
12        Integer intObj1 = 100;  
13        Integer intObj2 = 100;  
14        Integer intObj3 = new Integer(100);  
15        Integer intObj4 = new Integer(200);  
16  
17        // Using == with Integer objects (within -128 to 127 range)  
18        System.out.println("intObj1 == intObj2: " + (intObj1 == intObj2));  
19    }  
}
```

Problems Javadoc Declaration Console X  
<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre1  
intObj1 == intObj2: true

3) `intObj1 == intObj3`:

Result: False

```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5
6         //Mention what's the result in 1, 2, 3, 4 and 5
7         // Primitive int
8         int int1 = 100;
9         int int2 = 100;
10
11         // Integer objects
12         Integer intObj1 = 100;
13         Integer intObj2 = 100;
14         Integer intObj3 = new Integer(100);
15         Integer intObj4 = new Integer(200);
16
17         // Using == with Integer objects (new instance)
18         System.out.println("intObj1 == intObj3: " + (intObj1 == intObj3));
19     }
20 }
```

< Problems Javadoc Declaration Console X

<terminated> operators [Java Application] C:\Users\swathishree.s.p\p2\pool\plugins\org.eclipse.justi.openjdk.hot  
intObj1 == intObj3: false

4) `intObj1.equals(intObj3):`

Result: True

```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5
6         //Mention what's the result in 1, 2, 3,4 and 5
7         // Primitive int
8         int int1 = 100;
9         int int2 = 100;
10
11         // Integer objects
12         Integer intObj1 = 100;
13         Integer intObj2 = 100;
14         Integer intObj3 = new Integer(100);
15         Integer intObj4 = new Integer(200);
16
17
18         // Using equals() with Integer objects
19         System.out.println("intObj1.equals(intObj3): " + intObj1.equals(intObj3));
20     }}
```

5) intObj1.equals(intObj4):

Result: False



```
1 package sample;
2
3 public class operators {
4     public static void main(String[] args) {
5
6         //Mention what's the result in 1, 2, 3,4 and 5
7         // Primitive int
8         int int1 = 100;
9         int int2 = 100;
10
11         // Integer objects
12         Integer intObj1 = 100;
13         Integer intObj2 = 100;
14         Integer intObj3 = new Integer(100);
15         Integer intObj4 = new Integer(200);
16
17         System.out.println("intObj1.equals(intObj4): " + intObj1.equals(intObj4));
18     }
19 }
```

Problems Javadoc Declaration Console X  
<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.  
intObj1.equals(intObj4): false

### CASE: 3

### QUESTION: 3

```
import java.io.BufferedReader;
```

```
import java.io.FileReader;
```

```
import java.io.IOException;
```

```
public class TryWithResourcesExample {
```

```
//Eliminating finally block to close resources.
```

```
    public static void main(String[] args) {
```

```
        // File path (adjust the path as needed)
```

```
        String filePath = "example.txt";
```

```
        // Traditional try-with-resources block
```

```
        try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {
```

```
            String line;
```

```
            while ((line = reader.readLine()) != null) {
```

```
                System.out.println(line);
```

```
            }
```

```
        } catch (IOException e) {
```

```

        e.printStackTrace();
    }
}

```

## ANSWER:

### Automatic Resource Management:

The `BufferedReader` and `FileReader` resources are automatically closed when the `try-with-resources` block is exited. You do not need a `finally` block to close these resources manually, which is the traditional approach.

### Enhanced Code (Using `try-with-resources`):

```

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

public class TryWithResourcesExample {
    //Eliminating finally block to close resources.
    public static void main(String[] args) {
        // File path (adjust the path as needed)
        String filePath = "example.txt";
        try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

```
13     try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {
14         String line;
15         while ((line = reader.readLine()) != null) {
16             System.out.println(line);
17         }
18     } catch (IOException e) {
19         e.printStackTrace();
20     }
21 }
22 }
23 }
24 }
```

Problems Javadoc Declaration Console X

<terminated> CopyBytes [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (11 Aug 2024, 11:18:33 am - 11:18:33 am)  
Hello everyone!,Welcome to Java

## CASE: 4

### QUESTION: 4

**import java.util.HashSet;**

**import java.util.LinkedHashSet;**

**import java.util.Set;**

**import java.util.TreeSet;**

**public class SetExample {**

**public static void main(String[] args) {**

**// Set 1. What's the order of elements?**

**Set<String> hashSet = new HashSet<>();**

**hashSet.add("Banana");**

**hashSet.add("Apple");**

**hashSet.add("Orange");**

**hashSet.add("Grapes");**

**System.out.println("HashSet: " + hashSet);**

**// LinkedHashSet 2. What's the order of elements ?**

**Set<String> linkedHashSet = new LinkedHashSet<>();**

**linkedHashSet.add("Banana");**

**linkedHashSet.add("Apple");**

**linkedHashSet.add("Orange");**

**linkedHashSet.add("Grapes");**



```
System.out.println("LinkedHashSet: " + linkedHashSet);
```

```
// TreeSet 1. What's the order of elements ?
```

```
Set<String> treeSet = new TreeSet<>();
```

```
treeSet.add("Banana");
```

```
treeSet.add("Apple");
```

```
treeSet.add("Orange");
```

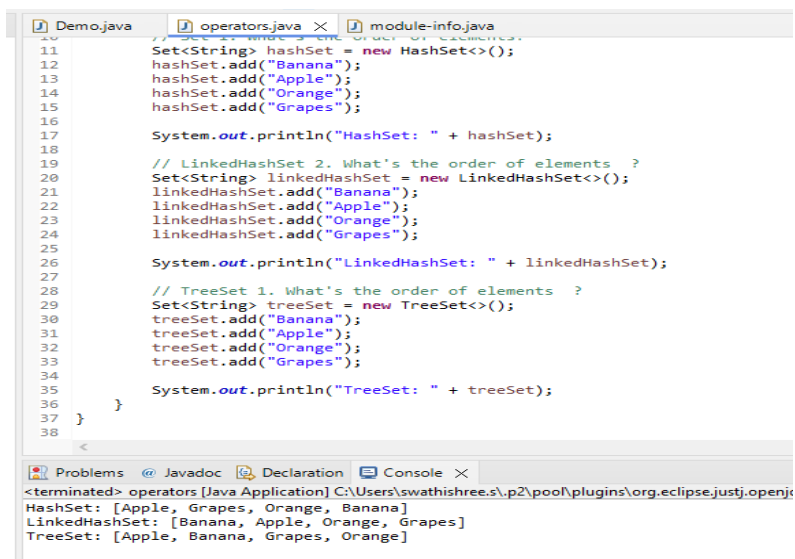
```
treeSet.add("Grapes");
```

```
System.out.println("TreeSet: " + treeSet);
```

```
}
```

```
}
```

**ANSWER:**



The screenshot shows an Eclipse IDE with a Java project named 'Demo.java'. The code in the editor is as follows:

```
11 Set<String> hashSet = new HashSet<>();
12 hashSet.add("Banana");
13 hashSet.add("Apple");
14 hashSet.add("Orange");
15 hashSet.add("Grapes");
16
17 System.out.println("HashSet: " + hashSet);
18
19 // LinkedHashSet 2. What's the order of elements ?
20 Set<String> linkedHashSet = new LinkedHashSet<>();
21 linkedHashSet.add("Banana");
22 linkedHashSet.add("Apple");
23 linkedHashSet.add("Orange");
24 linkedHashSet.add("Grapes");
25
26 System.out.println("LinkedHashSet: " + linkedHashSet);
27
28 // TreeSet 1. What's the order of elements ?
29 Set<String> treeSet = new TreeSet<>();
30 treeSet.add("Banana");
31 treeSet.add("Apple");
32 treeSet.add("Orange");
33 treeSet.add("Grapes");
34
35 System.out.println("TreeSet: " + treeSet);
36 }
37 }
38
```

The console output at the bottom shows the following results:

```
<terminated> operators [Java Application] C:\Users\swathishree.s\p2\pool\plugins\org.eclipse.justj.openj
HashSet: [Apple, Grapes, Orange, Banana]
LinkedHashSet: [Banana, Apple, Orange, Grapes]
TreeSet: [Apple, Banana, Grapes, Orange]
```