1.What is the default value of array for different data types?

Sol-

In Java, the default values of array elements depend on the type of data the array holds. Here’s a breakdown based on different data types:

Numeric Types:

Byte: 0

Short: 0

Int: 0

Long: 0L

Float: 0.0f

Double: 0.0d

Boolean Type:

Boolean: false

Character Type:

Char: The default value is the null character ‘\u0000’ (which is the Unicode character with value 0).

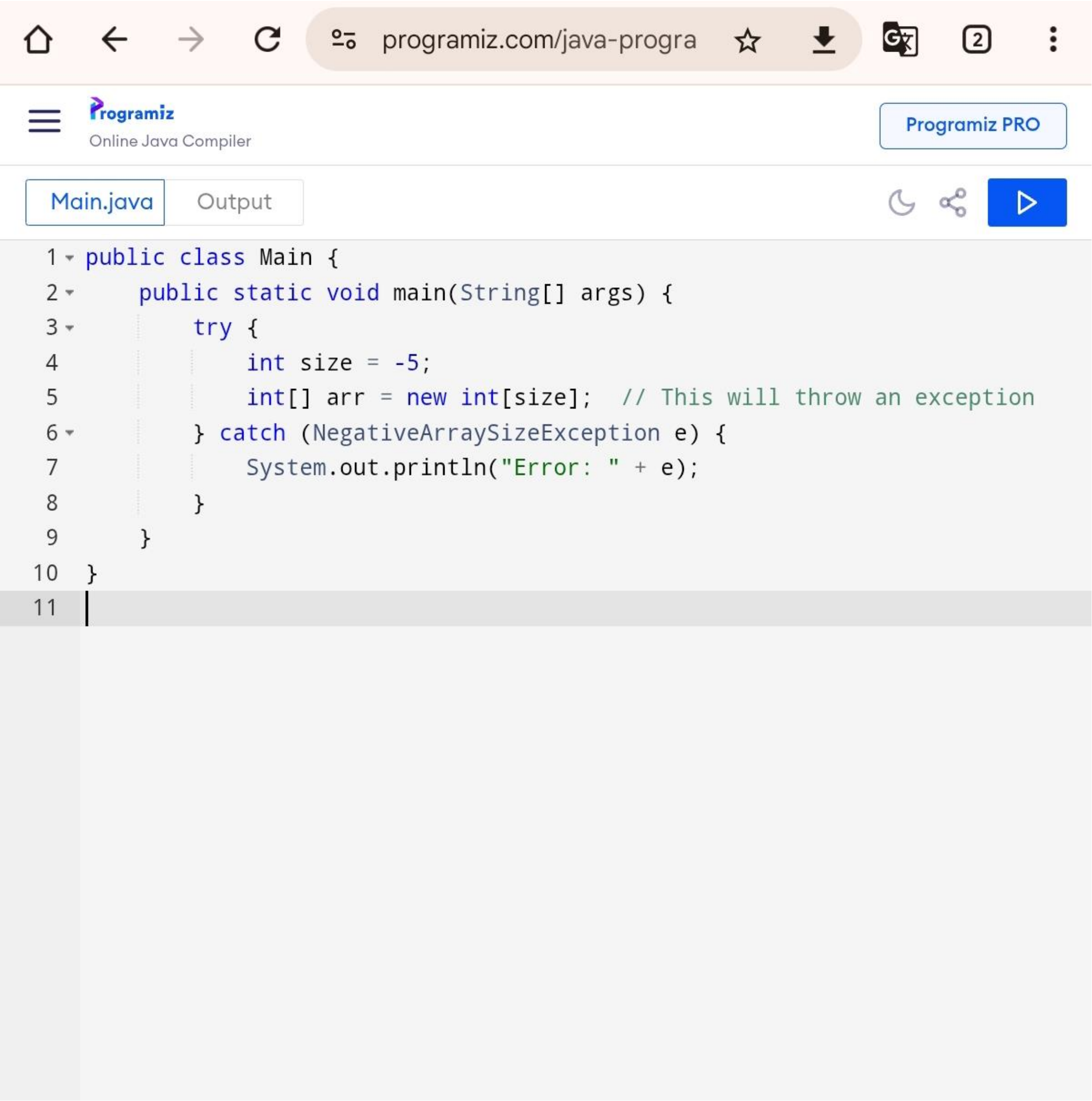
Object References (for non-primitive types):

Objects (like String, Integer, etc.): The default value is null.

2.**Can u pass negetive number in the array size ?**

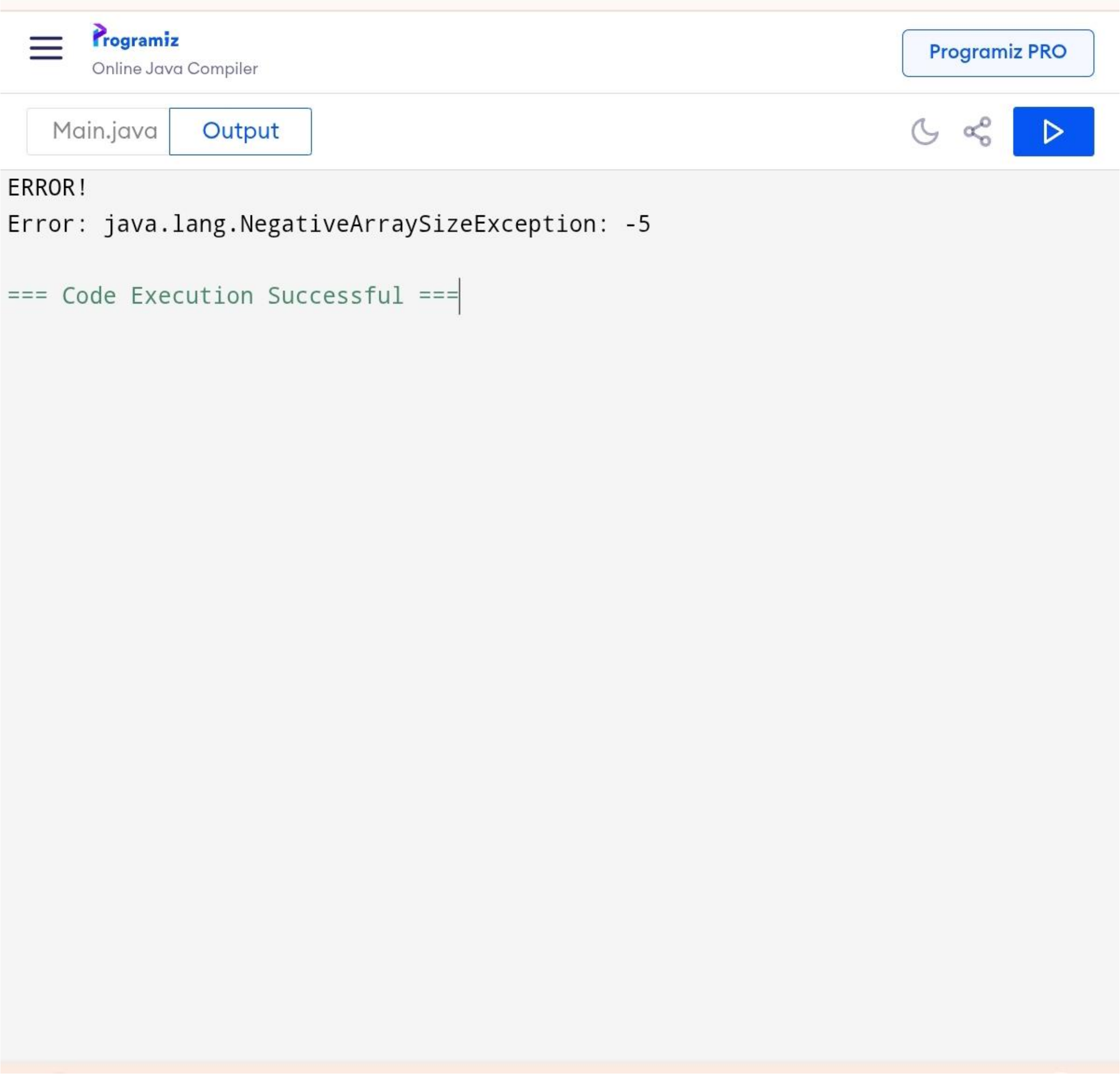
In Java, you cannot pass a negative number as the size of an array. When you try to do so, Java will throw a NegativeArraySizeException.

Example through code SCREENSHOT



```java
public class Main {
    public static void main(String[] args) {
        try {
            int size = -5;
            int[] arr = new int[size];  // This will throw an exception
        } catch (NegativeArraySizeException e) {
            System.out.println("Error: " + e);
        }
    }
}
```

OUTPUT screenshot



```
ERROR!
Error: java.lang.NegativeArraySizeException: -5

=== Code Execution Successful ===
```

3. Where does array stored in JVM memory ?

Sol-

In the Java Virtual Machine (JVM), arrays are objects and are stored in the heap memory. Here’s how it works:

Heap Memory: When you create an array in Java (e.g., int[] arr = new int[10];), the array itself is stored in the heap. The heap is a region of memory used for dynamic memory allocation where objects, including arrays, are stored. Even though arrays are a specialized type of object, they are still treated like objects by the JVM.

Array Elements: The elements of the array (e.g., the integers in the int[] array) are also stored in the heap. If the array is a reference type (such as String[]), the references to the actual objects are stored in the array, but the objects themselves are stored in the heap as well.

Stack Memory: The reference to the array, which is a pointer to the array object in the heap, is stored in stack memory if the array is declared in a method. If the array is an instance variable, its reference is stored in the heap as part of the object.

4.**What are the disadvantages of array?**

Sol

Arrays are widely used in programming because they offer efficient storage and access to elements. However, they also come with some disadvantages:

1. **Fixed Size:**

Description: Once an array is created with a specific size, it cannot be resized. If you need more elements, you must create a new array and copy the old elements into it.

Disadvantage: This can be inefficient, especially if the array size is not known in advance or changes frequently. It can also lead to wasted memory or require frequent resizing.

2. **Inefficient for Insertion and Deletion:**

Description: Inserting or deleting elements in the middle of an array can be inefficient, as it requires shifting elements to maintain the array’s order.

Disadvantage: These operations (insertion, deletion) have a time complexity of O(n), where n is the number of elements in the array. For large arrays, this can lead to significant performance issues.

5. What is an anonymous array in java ? Give example with code?

Sol-

An anonymous array in Java refers to an array that is created without explicitly assigning a variable name to it. It is commonly used when you need to pass an array as an argument to a method or when you don't need to reference the array after it's created.

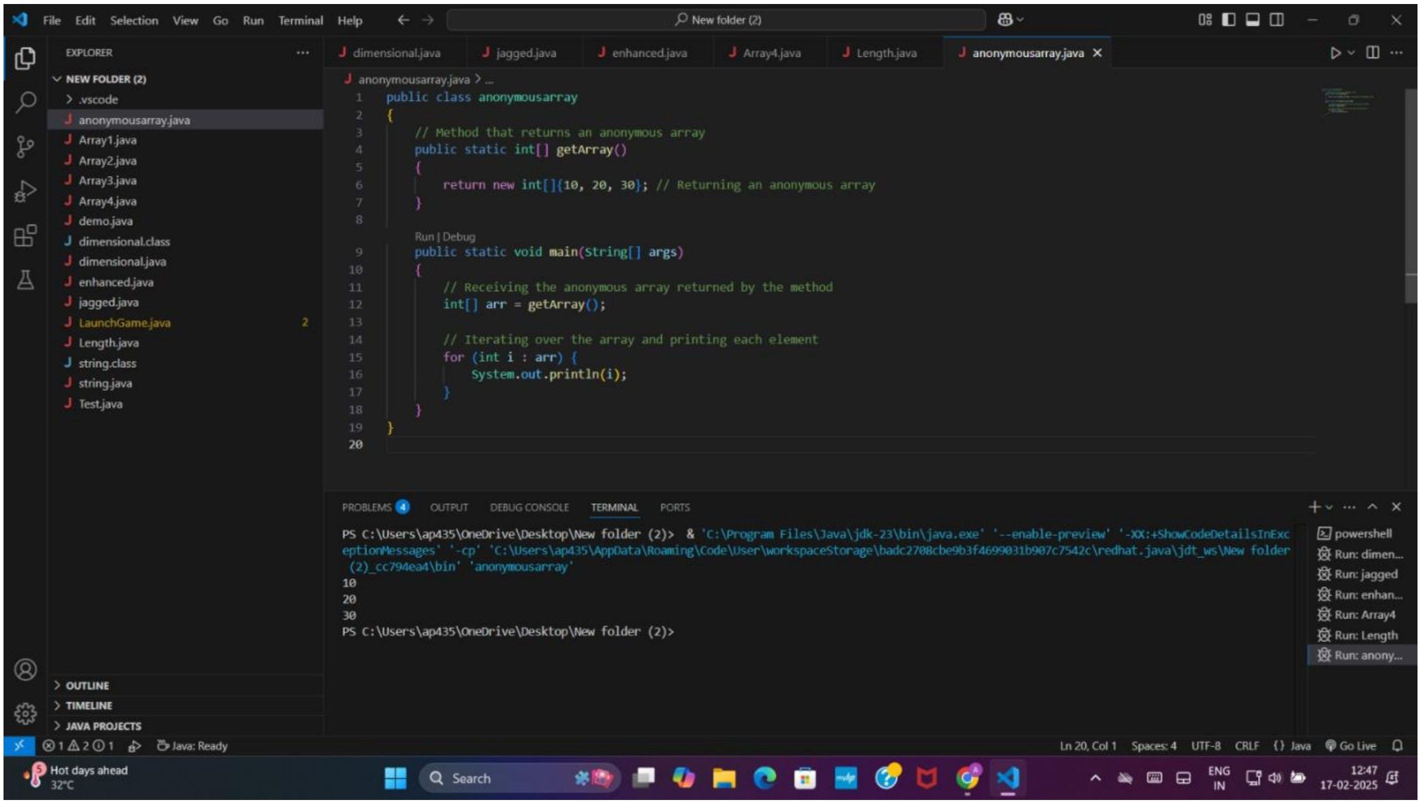
Key Points:

Anonymous arrays are created on-the-fly without storing them in a variable.

They are typically used when passing arrays as arguments to methods.

These arrays can be created inline, usually for temporary use.

Code as Example



```java
public class anonymousarray
{
    // Method that returns an anonymous array
    public static int[] getArray()
    {
        return new int[]{10, 20, 30}; // Returning an anonymous array
    }

    public static void main(String[] args)
    {
        // Receiving the anonymous array returned by the method
        int[] arr = getArray();

        // Iterating over the array and printing each element
        for (int i : arr) {
            System.out.println(i);
        }
    }
}
```

```
PS C:\Users\ap435\OneDrive\Desktop\New folder (2)>  & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ap435\AppData\Roaming\Code\User\workspaceStorage\badc2708cbe9b3f4699031b907c7542c\redhat.java\jdt_ws\New folder (2)_cc794ea4\bin' 'anonymousarray'
10
20
30
PS C:\Users\ap435\OneDrive\Desktop\New folder (2)>
```

6.What are the different ways to traverse an array in java?

Sol-

In Java, there are several ways to traverse or iterate over an array. Here are the most common methods:

1. **Using a for Loop**

The traditional for loop is one of the most common ways to traverse an arraay

Int[] arr = {1, 2, 3, 4, 5};

For (int I = 0; I < arr.length; i++)

{

```
  System.out.println(arr[i]);
}
```

2. Using an Enhanced for Loop (for-each loop)

This is a more concise way to iterate over an array

```
Int[] arr = {1, 2, 3, 4, 5};
For (int num : arr)
 {
  System.out.println(num);
}
```

7. What is the difference between length() and length in java?

Sol-

In Java, the difference between length and length() lies in how they are used for arrays and strings. Here's a detailed explanation and example:

1. Length (Property):

For arrays, length is a property (not a method). It is used to get the size of the array. It does not require parentheses when accessing it.

Example (for arrays):

```
Public class Main
{
  Public static void main(String[] args)
{
    Int[] arr = {1, 2, 3, 4, 5};
    System.out.println(arr.length);  // Output: 5 (size of the array)
  }
}
```

Usage: arr.length returns the number of elements in the array. It’s a fixed property of the array and is not a method, so no parentheses are used.

2. Length() (Method):

For strings, length() is a method. You need to call it with parentheses, even though it doesn’t take any arguments. It returns the number of characters in the string.

Examples

```
Public class Main

{

   Public static void main(String[] args)

 {

      String str = “Hello”;

      System.out.println(str.length());  // Output: 5

   }

}
```

Here, str.length() returns the number of characters in the string “Hello” (5).