

PATTERN MATCHING

Pattern to be matched:

Input will be an integer

Expected Output :

For n=3

1 2 3

7 8 9

4 5 6

For n=4

1 2 3 4

9 10 11 12

13 14 15 16

5 6 7 8

For n=5

1 2 3 4 5

11 12 13 14 15

21 22 23 24 25

16 17 18 19 20

6 7 8 9 10

Program written:

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);

        System.out.print("enter the number: ");

        int n = sc.nextInt();

        generatePattern(n);

    }

    public static void generatePattern(int n) {

        int[][] matrix = new int[n][n];

        int num = 1;

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

            int row;

            if (i % 2 == 0) {

                row = i / 2;

            } else {

                row = n - 1 - i / 2;

            }

        }

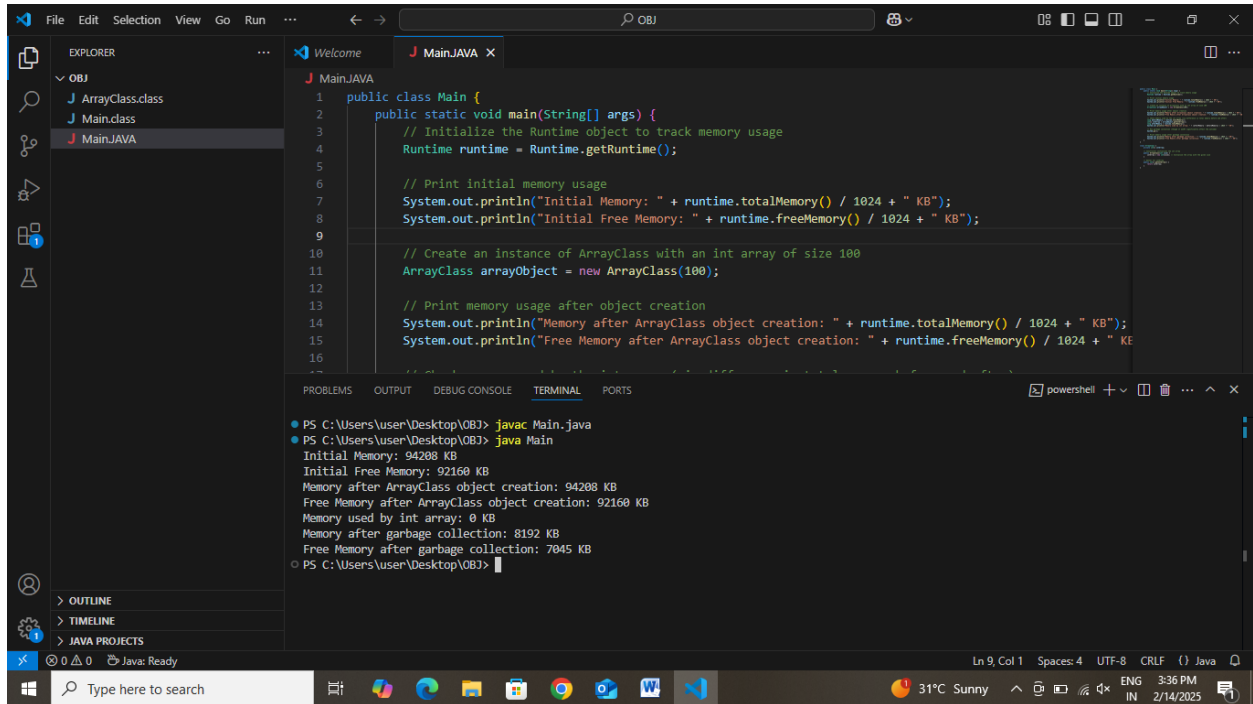
    }

}
```

```
        for (int j = 0; j < n; j++) {  
            matrix[row][j] = num++;  
        }  
    }  
  
    printMatrix(matrix, n);  
}  
  
public static void printMatrix(int[][] matrix, int n) {  
    for (int i = 0; i < n; i++) {  
        for (int j = 0; j < n; j++) {  
            System.out.print(matrix[i][j] + "\\t");  
        }  
        System.out.println();  
    }  
}  
}
```

MEMORY ALLOCATION WITH RESPECT TO OBJECTS

Array: int

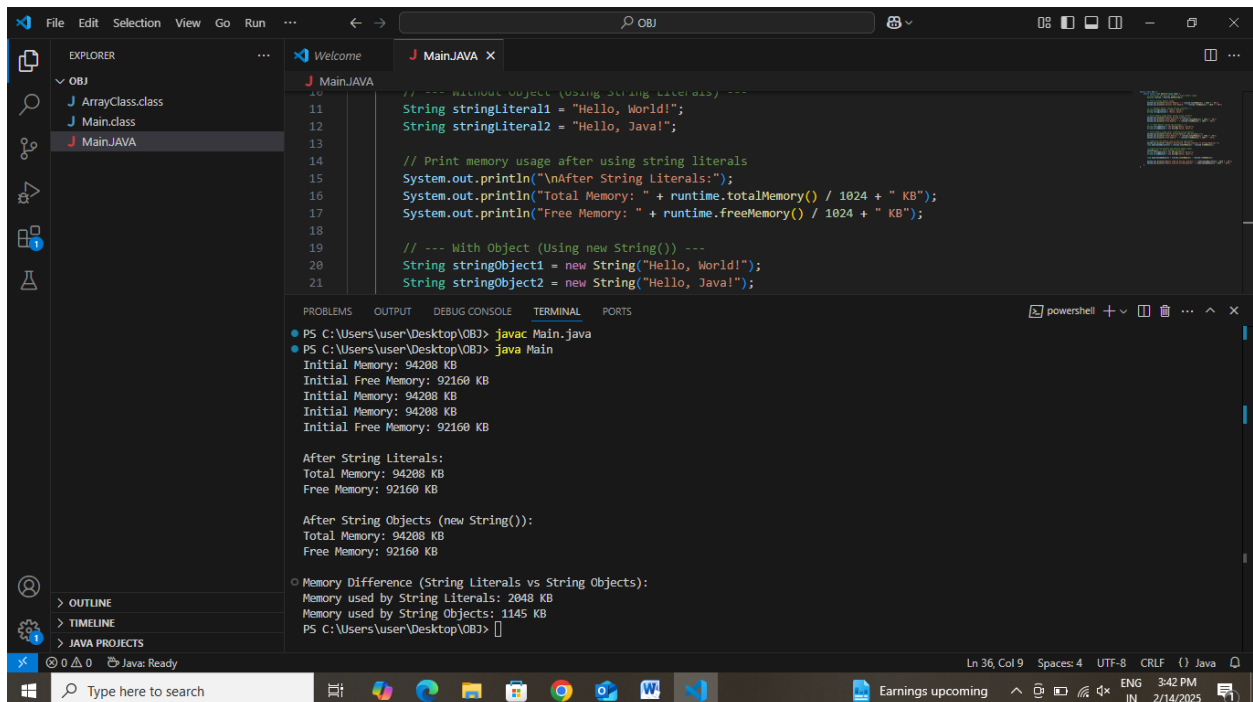


```
1 public class Main {
2     public static void main(String[] args) {
3         // Initialize the Runtime object to track memory usage
4         Runtime runtime = Runtime.getRuntime();
5
6         // Print initial memory usage
7         System.out.println("Initial Memory: " + runtime.totalMemory() / 1024 + " KB");
8         System.out.println("Initial Free Memory: " + runtime.freeMemory() / 1024 + " KB");
9
10        // Create an instance of ArrayClass with an int array of size 100
11        ArrayClass arrayObject = new ArrayClass(100);
12
13        // Print memory usage after object creation
14        System.out.println("Memory after ArrayClass object creation: " + runtime.totalMemory() / 1024 + " KB");
15        System.out.println("Free Memory after ArrayClass object creation: " + runtime.freeMemory() / 1024 + " KB");
16    }
17 }
```

Terminal Output:

```
PS C:\Users\user\Desktop\OBJ> javac Main.java
PS C:\Users\user\Desktop\OBJ> java Main
Initial Memory: 94288 KB
Initial Free Memory: 92160 KB
Memory after ArrayClass object creation: 94288 KB
Free Memory after ArrayClass object creation: 92160 KB
Memory used by int array: 0 KB
Memory after garbage collection: 8192 KB
Free Memory after garbage collection: 7945 KB
PS C:\Users\user\Desktop\OBJ>
```

Array : String(with object and without object comparission)



```
11 String stringLiteral1 = "Hello, World!";
12 String stringLiteral2 = "Hello, Java!";
13
14 // Print memory usage after using string literals
15 System.out.println("\nAfter String Literals:");
16 System.out.println("Total Memory: " + runtime.totalMemory() / 1024 + " KB");
17 System.out.println("Free Memory: " + runtime.freeMemory() / 1024 + " KB");
18
19 // --- With Object (Using new String()) ---
20 String stringObject1 = new String("Hello, World!");
21 String stringObject2 = new String("Hello, Java!");
```

Terminal Output:

```
PS C:\Users\user\Desktop\OBJ> javac Main.java
PS C:\Users\user\Desktop\OBJ> java Main
Initial Memory: 94288 KB
Initial Free Memory: 92160 KB
Initial Memory: 94288 KB
Initial Free Memory: 92160 KB

After String Literals:
Total Memory: 94288 KB
Free Memory: 92160 KB

After String Objects (new String()):
Total Memory: 94288 KB
Free Memory: 92160 KB

Memory Difference (String Literals vs String Objects):
Memory used by String Literals: 2048 KB
Memory used by String Objects: 1145 KB
PS C:\Users\user\Desktop\OBJ>
```