

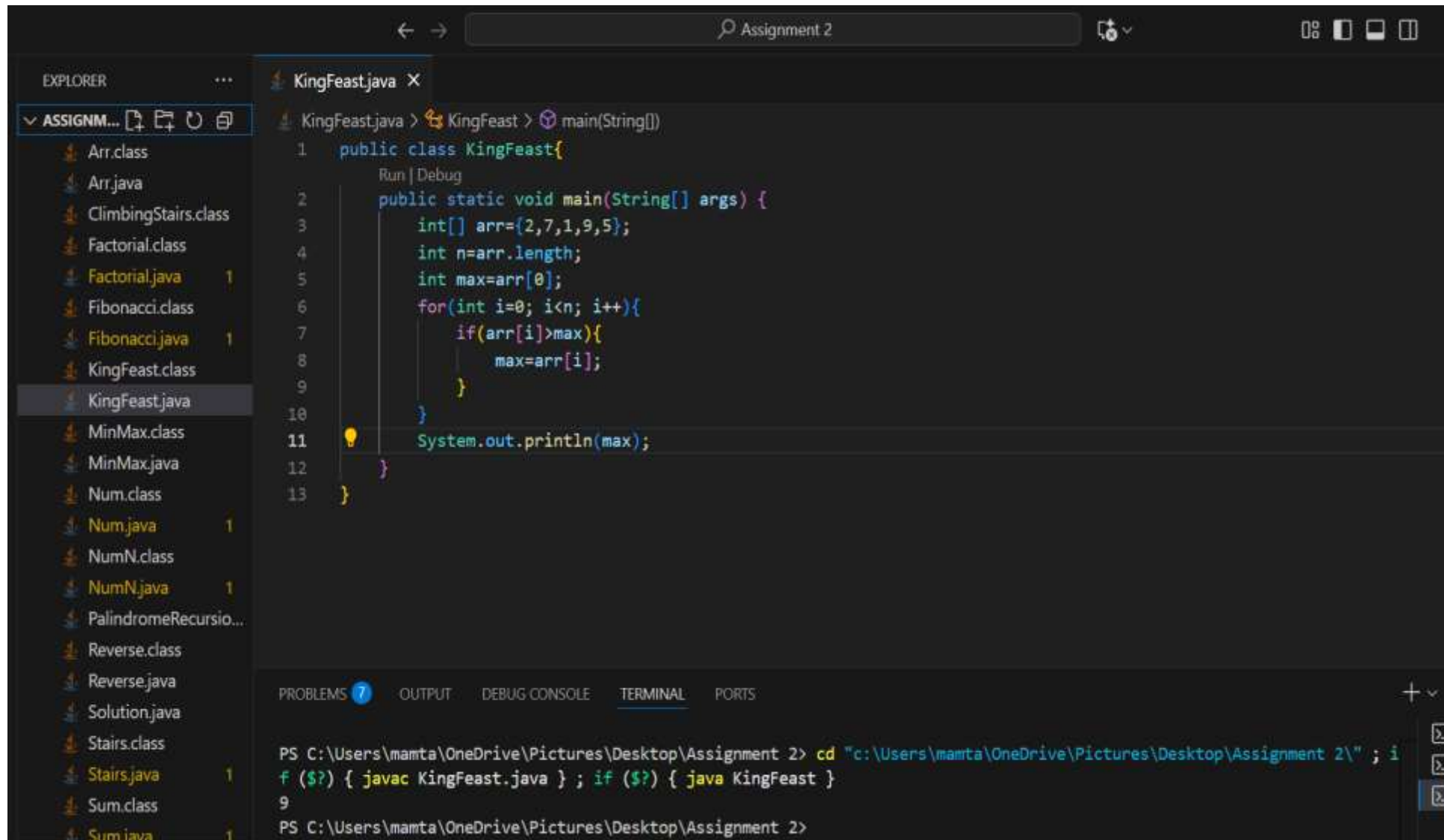
# Assignment-3

DSA-Problems

(Date: 06 October, 2025)

- Name:- Abhay Fulara
- Admission no:- 24SCSE1180211
- Section:- 34

# Q1. The King's Feast



The screenshot shows an IDE with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The file explorer lists various Java files, with 'KingFeast.java' selected. The code editor displays the following Java code:

```
KingFeast.java > KingFeast > main(String[])
1 public class KingFeast{
2     public static void main(String[] args) {
3         int[] arr={2,7,1,9,5};
4         int n=arr.length;
5         int max=arr[0];
6         for(int i=0; i<n; i++){
7             if(arr[i]>max){
8                 max=arr[i];
9             }
10        }
11        System.out.println(max);
12    }
13 }
```

The terminal at the bottom shows the command prompt and the execution of the program:

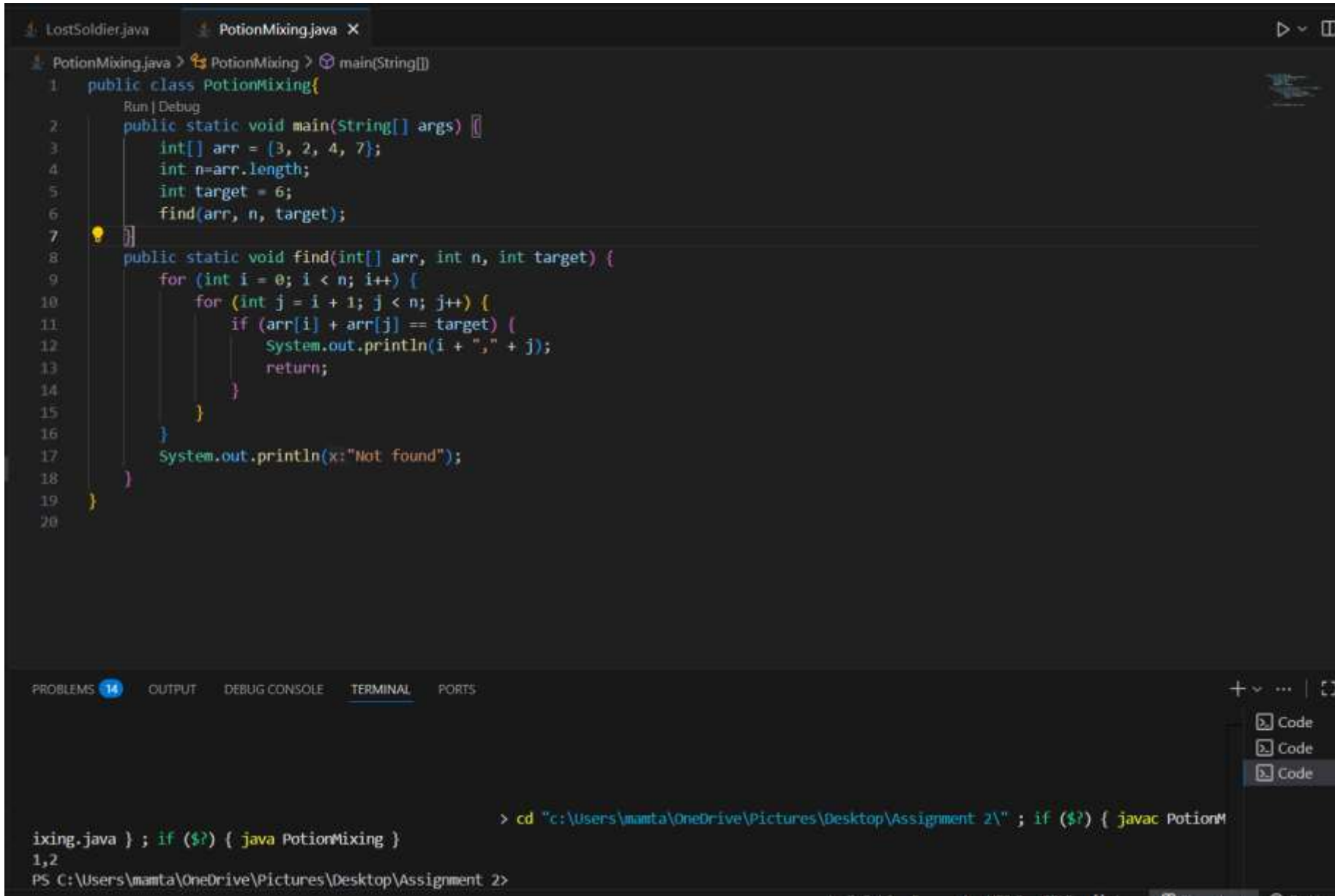
```
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; i
f ($?) { javac KingFeast.java } ; if ($?) { java KingFeast }
9
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q2. The Lost Soldier

```
LostSoldier.java X
LostSoldier.java > LostSoldier > find(int[], int)
1 public class LostSoldier {
2     Run | Debug
3     public static void main(String[] args) {
4         int n = 5;
5         int[] arr = {0, 1, 2, 4, 5};
6
7         int lost= find(arr, n);
8         System.out.println(lost);
9     }
10
11     public static int find(int[] arr, int n) {
12         int totalSum = n * (n + 1) / 2;
13         int actualSum = 0;
14
15         for (int i = 0; i < arr.length; i++) {
16             actualSum += arr[i];
17         }
18
19         return totalSum - actualSum;
20     }
21 }

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac LostSol
dier.java } ; if ($?) { java LostSoldier }
3
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

### Q3. Potion Mixing (Two Sum)



The screenshot shows an IDE with two tabs: `LostSoldier.java` and `PotionMixing.java`. The `PotionMixing.java` tab is active, displaying the following Java code:

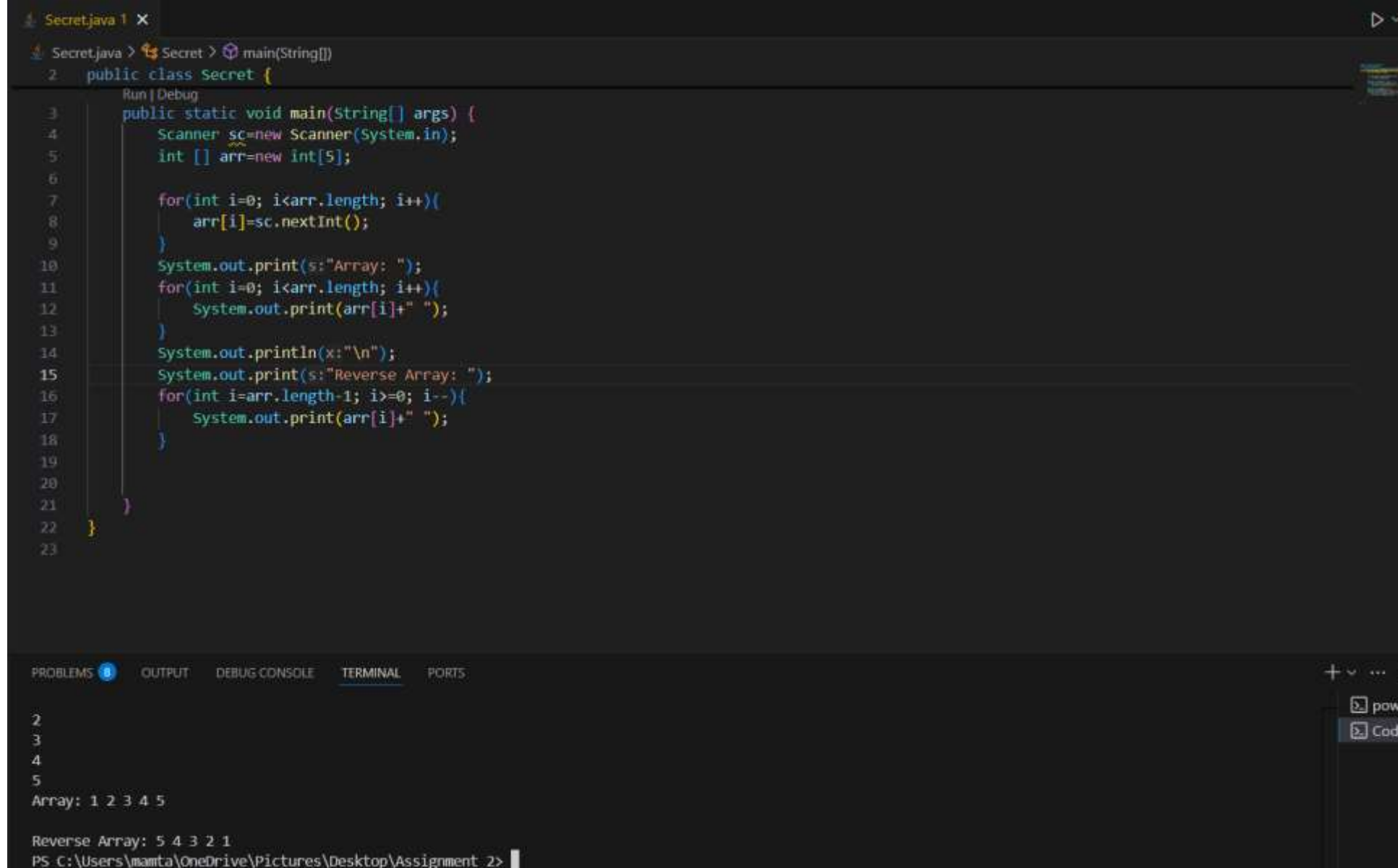
```
1 public class PotionMixing{
2     public static void main(String[] args) {
3         int[] arr = {3, 2, 4, 7};
4         int n=arr.length;
5         int target = 6;
6         find(arr, n, target);
7     }
8     public static void find(int[] arr, int n, int target) {
9         for (int i = 0; i < n; i++) {
10             for (int j = i + 1; j < n; j++) {
11                 if (arr[i] + arr[j] == target) {
12                     System.out.println(i + "," + j);
13                     return;
14                 }
15             }
16         }
17         System.out.println("Not found");
18     }
19 }
20
```

The IDE interface includes a sidebar on the left with tabs for `PROBLEMS` (14), `OUTPUT`, `DEBUG CONSOLE`, `TERMINAL`, and `PORTS`. The `TERMINAL` tab is active, showing the following commands and output:

```
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac PotionM
ixing.java } ; if ($?) { java PotionMixing }
1,2
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

On the right side of the IDE, there are three tabs labeled `Code`.

## Q4. The Secret Message



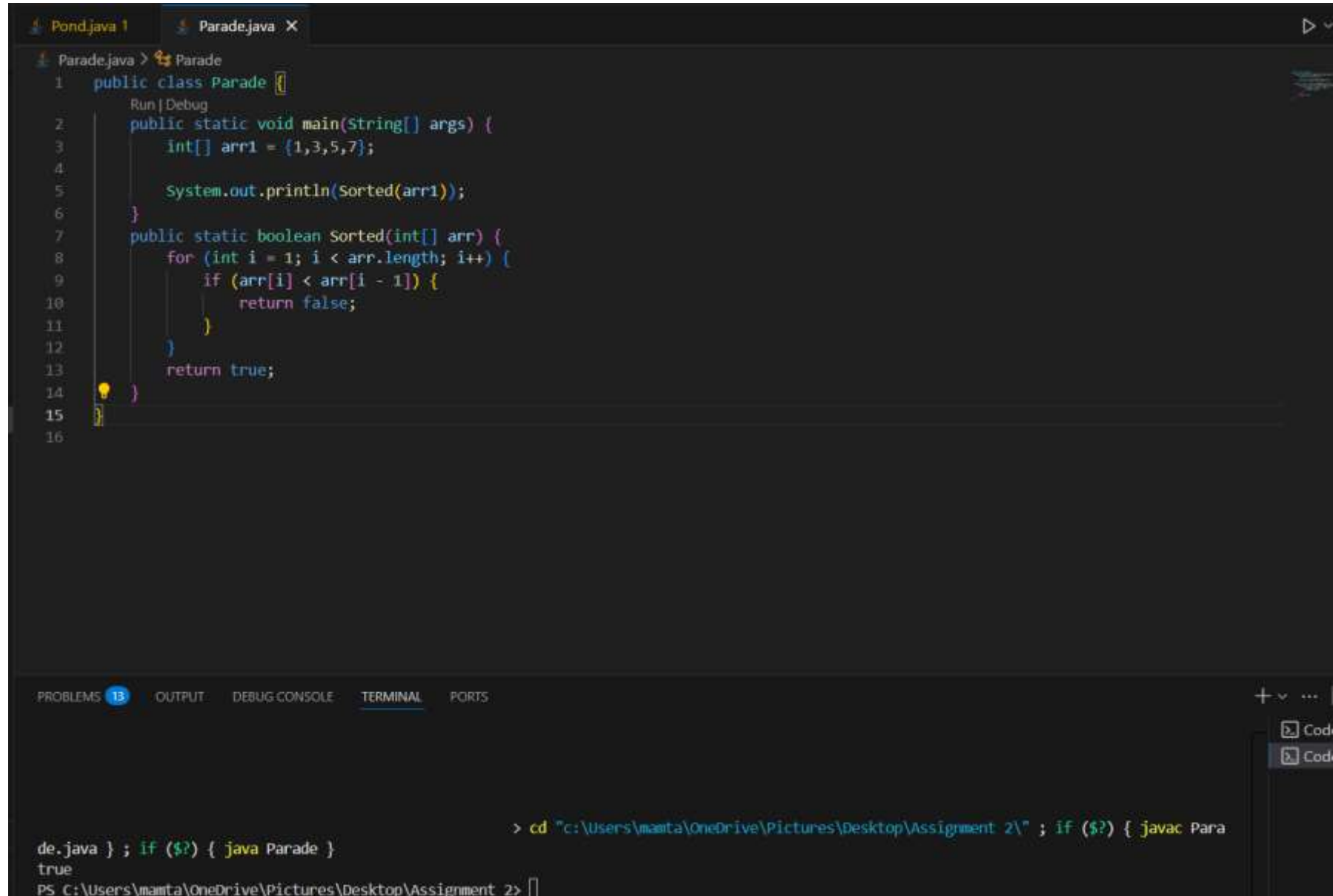
```
Secret.java 1 x
Secret.java > Secret > main(String[])
2 public class Secret {
    Run | Debug
3     public static void main(String[] args) {
4         Scanner sc=new Scanner(System.in);
5         int [] arr=new int[5];
6
7         for(int i=0; i<arr.length; i++){
8             arr[i]=sc.nextInt();
9         }
10        System.out.print(s:"Array: ");
11        for(int i=0; i<arr.length; i++){
12            System.out.print(arr[i]+" ");
13        }
14        System.out.println(x:"\n");
15        System.out.print(s:"Reverse Array: ");
16        for(int i=arr.length-1; i>=0; i--){
17            System.out.print(arr[i]+" ");
18        }
19
20    }
21 }
22
23
```

PROBLEMS 0 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
2
3
4
5
Array: 1 2 3 4 5

Reverse Array: 5 4 3 2 1
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q5. The King's Parade



```
Parade.java > Parade
1 public class Parade {
2     public static void main(String[] args) {
3         int[] arr1 = {1,3,5,7};
4
5         System.out.println(Sorted(arr1));
6     }
7     public static boolean Sorted(int[] arr) {
8         for (int i = 1; i < arr.length; i++) {
9             if (arr[i] < arr[i - 1]) {
10                 return false;
11             }
12         }
13         return true;
14     }
15 }
16

PROBLEMS 13 OUTPUT DEBUG CONSOLE TERMINAL PORTS

> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Para
de.java } ; if ($?) { java Parade }
true
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q6. The Treasure Island

```
TreasureIsland.java X
TreasureIsland.java > TreasureIsland > main(String[])
1  import java.util.Scanner;
2  public class TreasureIsland {
    Run | Debug
3      public static void main(String[] args) {
4          Scanner sc = new Scanner(System.in);
5          int n = sc.nextInt();
6          int m = sc.nextInt();
7          int[][] arr = new int[n][m];
8
9          for (int i = 0; i < n; i++) {
10             for (int j = 0; j < m; j++) {
11                 arr[i][j] = sc.nextInt();
12             }
13         }
14
15         int maxSum = 0;
16         int maxRow = 0;
17
18         for (int i = 0; i < n; i++) {
19             int sum = 0;
20             for (int j = 0; j < m; j++) {
21                 sum += arr[i][j];
22             }
23             if (sum > maxSum) {
24                 maxSum = sum;
25                 maxRow = i + 1;
26             }
27         }
28
29         System.out.println("Row " + maxRow + " (sum=" + maxSum + ")");
30         sc.close();
31     }
32 }
33
34
--
PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS
4
5
6
7
8
9
Row 3 (sum=24)
```

## Q7. The Spiral Library

```
TreasureIsland.java  SpiralLibrary.java 1 X
SpiralLibrary.java > SpiralLibrary > print(int[][], int, int)
2 public class SpiralLibrary {
17
18     public static void print(int[][] matrix, int rows, int cols) {
19         int top = 0, bottom = rows - 1;
20         int left = 0, right = cols - 1;
21
22         while (top <= bottom && left <= right) {
23             for (int j = left; j <= right; j++) {
24                 System.out.print(matrix[top][j] + " ");
25             }
26             top++;
27
28             for (int i = top; i <= bottom; i++) {
29                 System.out.print(matrix[i][right] + " ");
30             }
31             right--;
32
33             if (top <= bottom) {
34                 for (int j = right; j >= left; j--) {
35                     System.out.print(matrix[bottom][j] + " ");
36                 }
37                 bottom--;
38             }
39
40             if (left <= right) {
41                 for (int i = bottom; i >= top; i--) {
42                     System.out.print(matrix[i][left] + " ");
43                 }
44                 left++;
45             }
46         }
47     }
48 }
49
50
```

PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS Code

```
4
5
6
7
8
9
1 2 3 6 9 8 7 4 5
```



## Q8. The Royal Diagonal

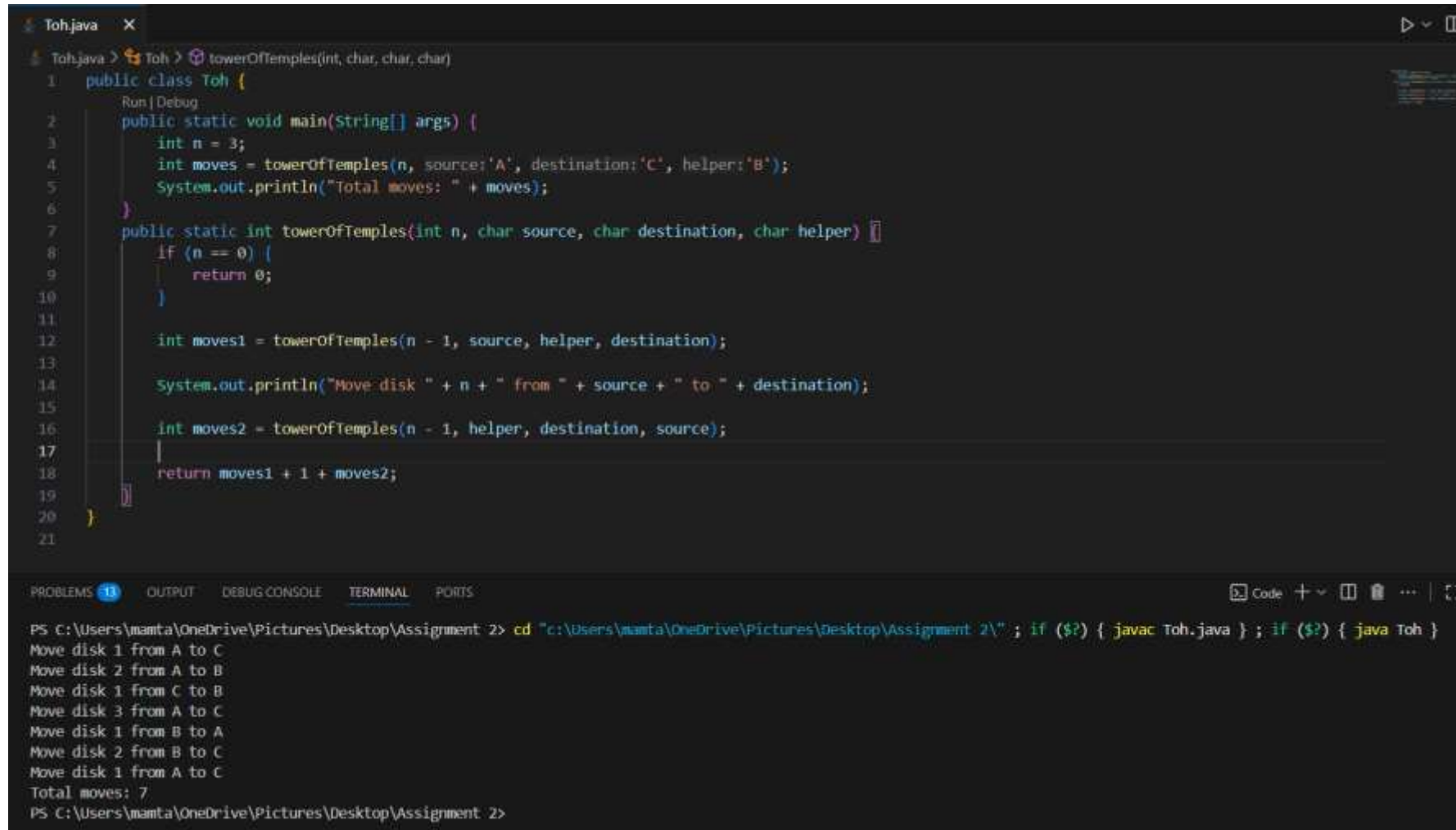
```
Diagonal.java 1 X
Diagonal.java > Diagonal > main(String[])
1  import java.util.Scanner;
2  public class Diagonal {
    Run | Debug
3      public static void main(String[] args) {
4          Scanner sc=new Scanner(System.in);
5          int n=sc.nextInt();
6          int m=sc.nextInt();
7          int [][] arr=new int[n][m];
8
9          for(int i=0; i<arr.length;i++){
10             for(int j=0;j<arr.length;j++){
11                 arr[i][j]=sc.nextInt();
12             }
13         }
14         System.out.println("Array: ");
15         for(int i=0; i<arr.length;i++){
16             for(int j=0;j<arr.length;j++){
17                 System.out.print(arr[i][j]+" ");
18             }
19             System.out.println();
20         }
21
22         int Diagonal1 = 0;
23         int Diagonal2 = 0;
24
25         for (int i = 0; i < n; i++) {
26             Diagonal1 += arr[i][i];
27             Diagonal2 += arr[i][m - 1 - i];
28         }
29         System.out.println(Diagonal1);
30         System.out.println(Diagonal2);
31
32     }
33 }
34
PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS
9
Array:
1 2 3
4 5 6
7 8 9
15
15
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q10. The Rainwater Pond

```
Pond.java X
Pond.java > Pond
1 import java.util.Scanner;
2 public class Pond {
3     public static void main(String[] args) {
4         Scanner sc=new Scanner(System.in);
5         int n=sc.nextInt();
6         int [][] arr=new int[n][n];
7
8         for(int i=0; i<arr.length;i++){
9             for(int j=0;j<arr.length;j++){
10                 arr[i][j]=sc.nextInt();
11             }
12         }
13         System.out.println("Array: ");
14         for(int i=0; i<arr.length;i++){
15             for(int j=0;j<arr.length;j++){
16                 System.out.print(arr[i][j]+" ");
17             }
18             System.out.println();
19         }
20
21         int sum=0;
22         for(int i=0; i<arr.length;i++){
23             for(int j=0;j<arr.length;j++){
24                 if(arr[i][j]==1){
25                     sum++;
26                 }
27             }
28         }
29         System.out.println(sum);
30
31
32
33
34 }

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS
0
1
Array:
1 0 1
0 1 0
1 0 1
5
PS C:\Users\maanta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q11. Tower of Temples (Hanoi)



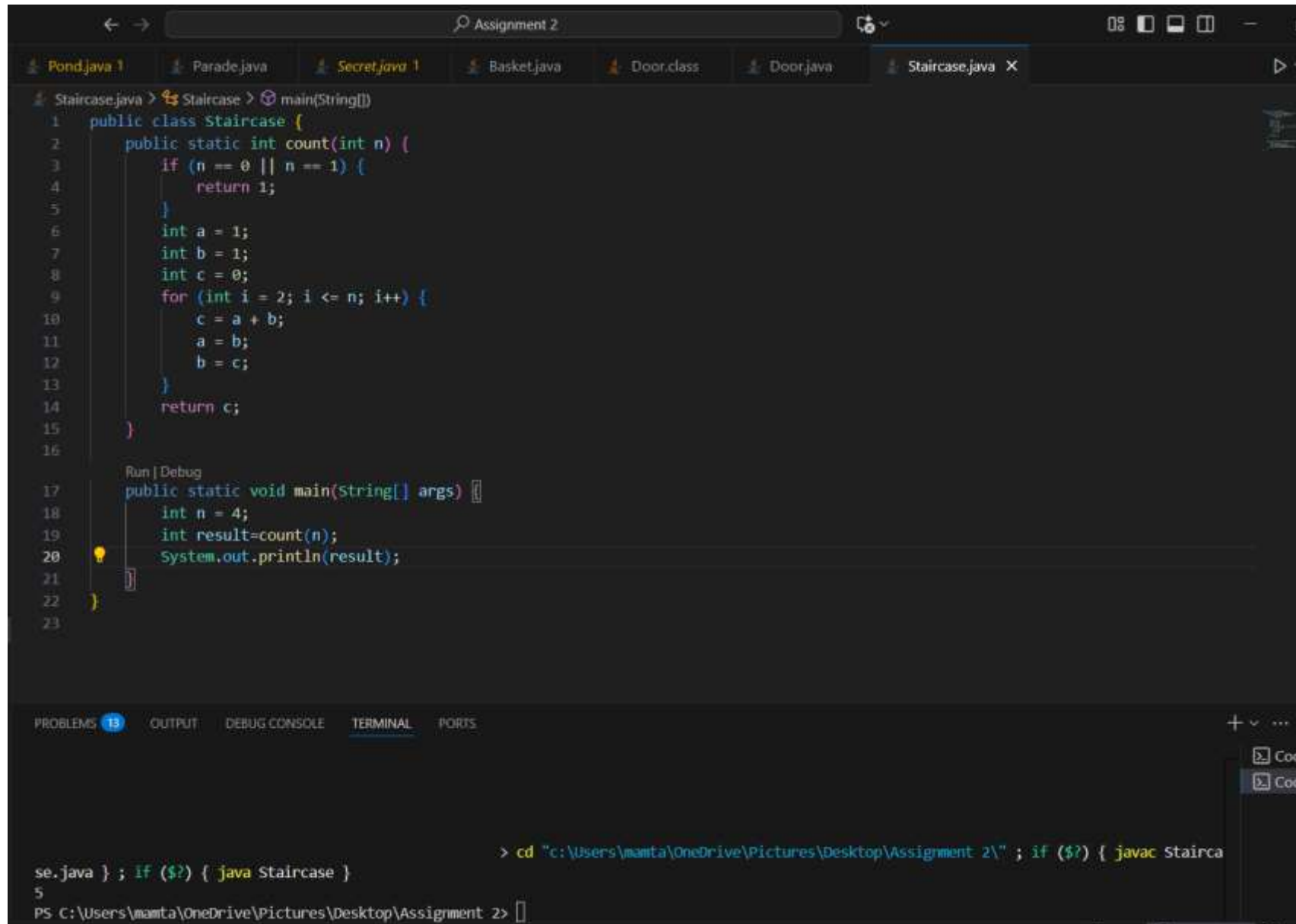
The screenshot shows an IDE window titled 'Toh.java'. The code defines a public class 'Toh' with a 'main' method and a recursive 'towerOfTemples' method. The 'main' method sets 'n' to 3 and calls 'towerOfTemples' with source 'A', destination 'C', and helper 'B'. The 'towerOfTemples' method uses recursion to calculate the minimum number of moves and prints the sequence of moves. The output window at the bottom shows the execution results, including the sequence of moves and the total count of 7 moves.

```
Toh.java
Toh.java > Toh > towerOfTemples(int, char, char, char)
1 public class Toh {
2     public static void main(String[] args) {
3         int n = 3;
4         int moves = towerOfTemples(n, source:'A', destination:'C', helper:'B');
5         System.out.println("Total moves: " + moves);
6     }
7     public static int towerOfTemples(int n, char source, char destination, char helper) {
8         if (n == 0) {
9             return 0;
10        }
11
12        int moves1 = towerOfTemples(n - 1, source, helper, destination);
13
14        System.out.println("Move disk " + n + " from " + source + " to " + destination);
15
16        int moves2 = towerOfTemples(n - 1, helper, destination, source);
17
18        return moves1 + 1 + moves2;
19    }
20 }
21
```

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS

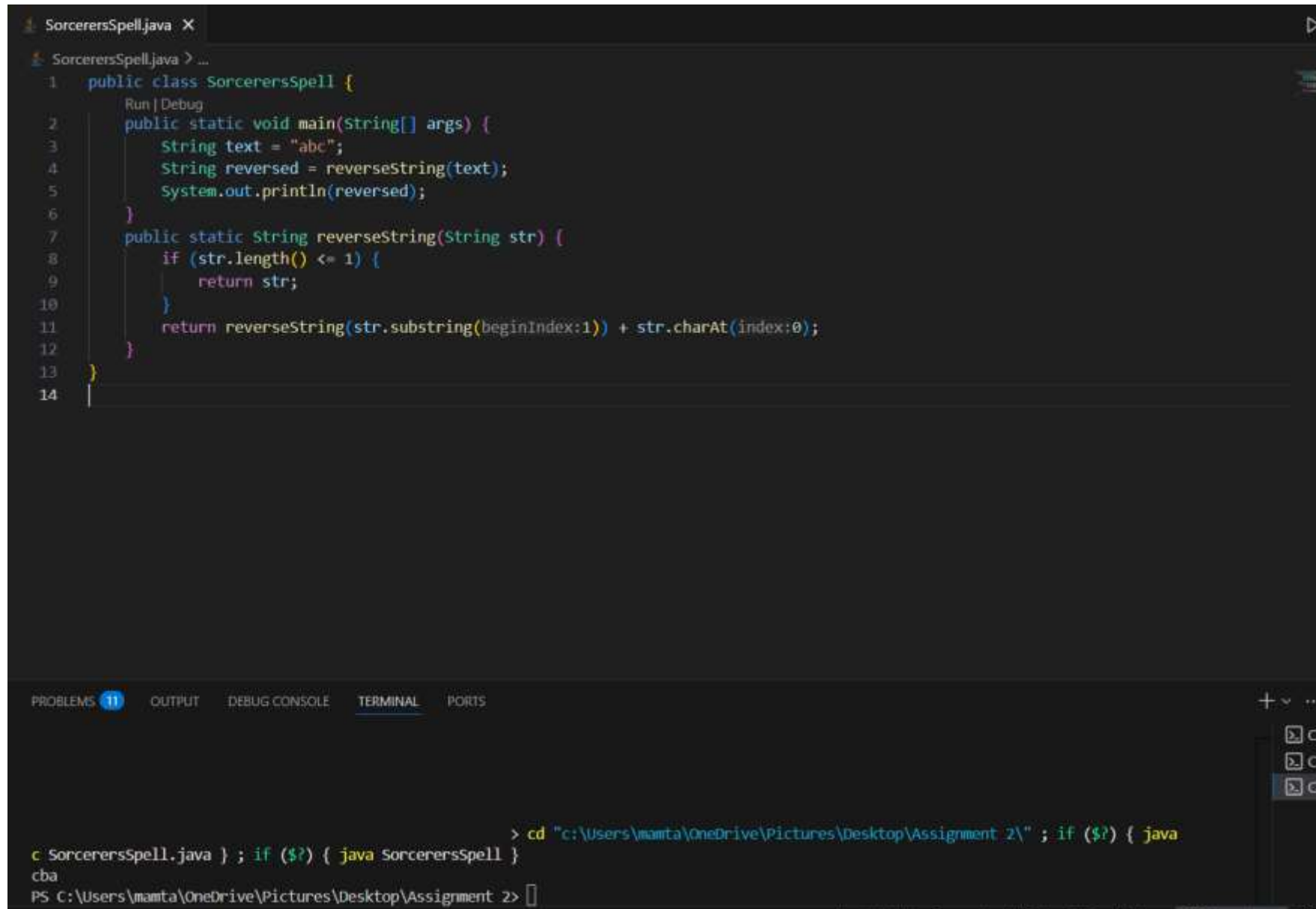
```
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Toh.java } ; if ($?) { java Toh }
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
Total moves: 7
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q12. The Magical Staircase



```
Assignment 2
Pond.java 1 Parade.java Secret.java 1 Basket.java Door.class Door.java Staircase.java X
Staircase.java > Staircase > main(String[])
1 public class Staircase {
2     public static int count(int n) {
3         if (n == 0 || n == 1) {
4             return 1;
5         }
6         int a = 1;
7         int b = 1;
8         int c = 0;
9         for (int i = 2; i <= n; i++) {
10            c = a + b;
11            a = b;
12            b = c;
13        }
14        return c;
15    }
16
17    Run | Debug
18    public static void main(String[] args) {
19        int n = 4;
20        int result=count(n);
21        System.out.println(result);
22    }
23
PROBLEMS 13 OUTPUT DEBUG CONSOLE TERMINAL PORTS
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Stairca
se.java } ; if ($?) { java Staircase }
5
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q13. The Sorcerer's Spell



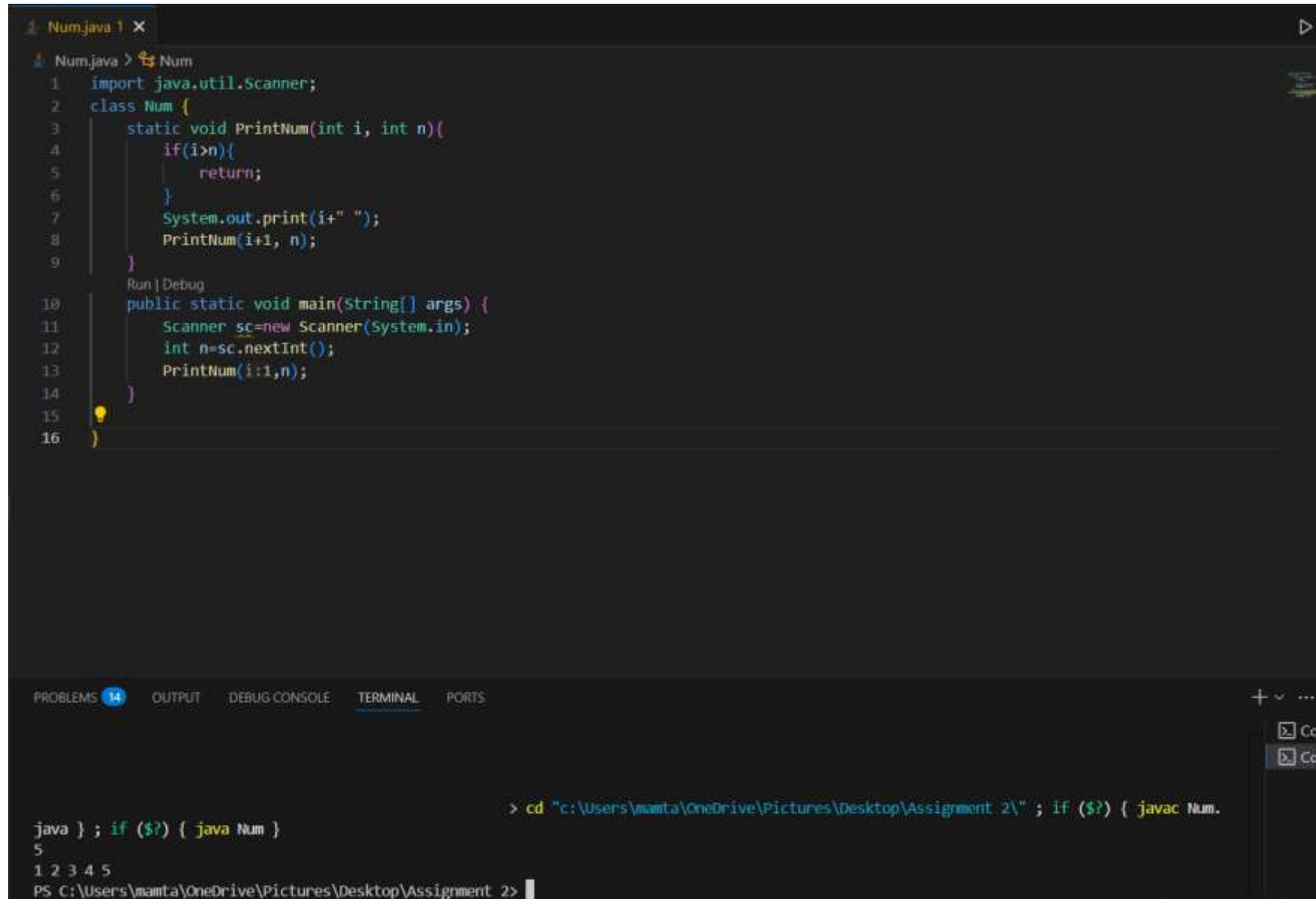
The screenshot shows an IDE with a file named `SorcerersSpell.java`. The code defines a class `SorcerersSpell` with a `main` method and a `reverseString` method. The `main` method initializes a string `text` to "abc", calls `reverseString`, and prints the result. The `reverseString` method uses recursion to reverse the string. The bottom panel shows the terminal with the command to run the program, which outputs "cba".

```
SorcerersSpell.java X
SorcerersSpell.java > ...
1 public class SorcerersSpell {
2     Run | Debug
3     public static void main(String[] args) {
4         String text = "abc";
5         String reversed = reverseString(text);
6         System.out.println(reversed);
7     }
8     public static String reverseString(String str) {
9         if (str.length() <= 1) {
10             return str;
11         }
12         return reverseString(str.substring(beginIndex:1)) + str.charAt(index:0);
13     }
14 }
```

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { java
c SorcerersSpell.java } ; if ($?) { java SorcerersSpell }
cba
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q14. The Dragon's Roar



The screenshot shows an IDE with a Java file named `Num.java`. The code defines a class `Num` with a static method `PrintNum` and a `main` method. The `PrintNum` method prints numbers from `i` to `n` if `i` is less than or equal to `n`, and returns if `i` is greater than `n`. The `main` method uses a `Scanner` to read an integer `n` and calls `PrintNum(1, n)`.

```
Num.java 1 X
Num.java > Num
1  import java.util.Scanner;
2  class Num {
3      static void PrintNum(int i, int n){
4          if(i>n){
5              return;
6          }
7          System.out.print(i+" ");
8          PrintNum(i+1, n);
9      }
10
11     public static void main(String[] args) {
12         Scanner sc=new Scanner(System.in);
13         int n=sc.nextInt();
14         PrintNum(1,n);
15     }
16 }
```

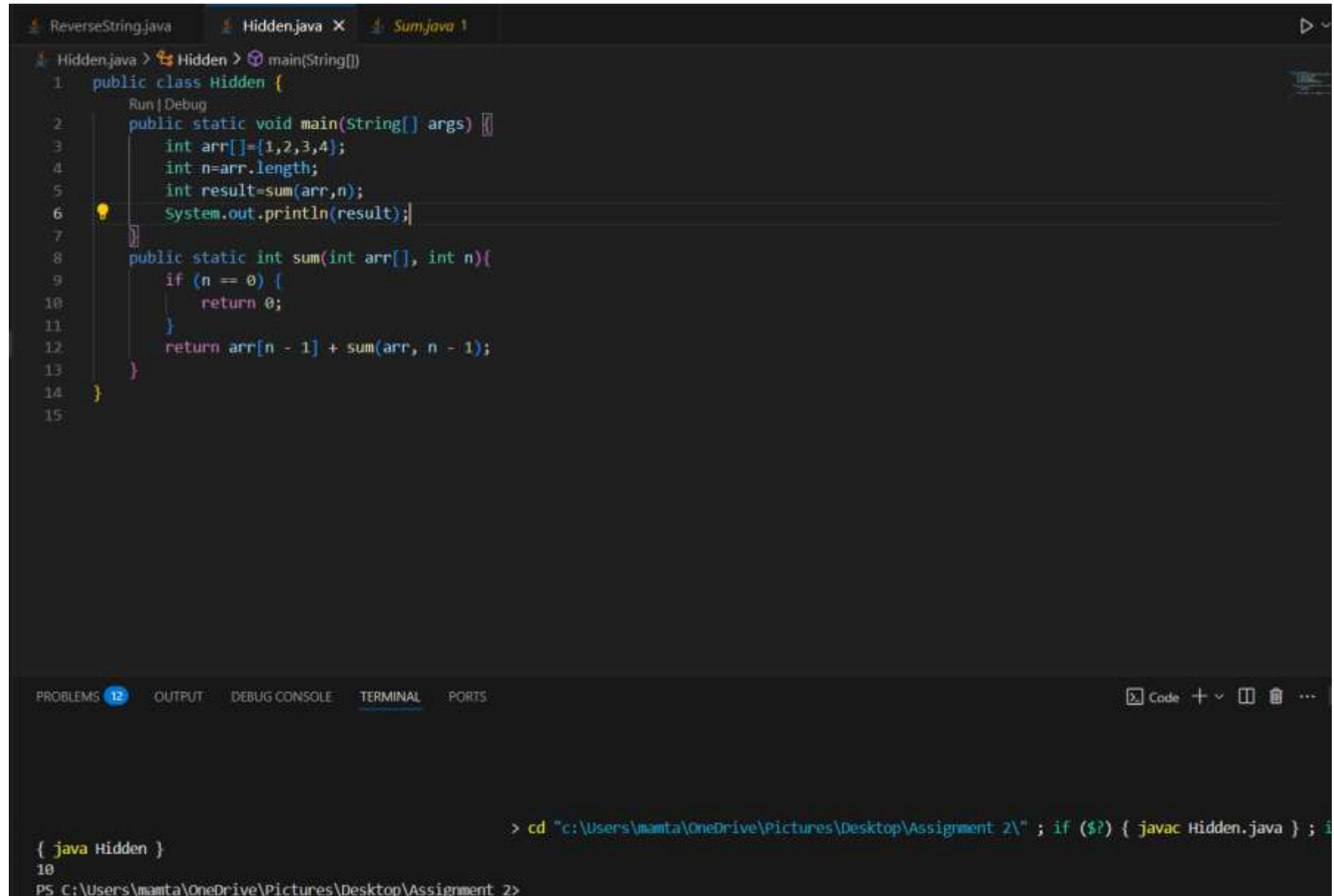
The bottom panel shows the terminal output and command prompt:

```
java } ; if ($?) { java Num }
5
1 2 3 4 5
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

Terminal command history:

```
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Num.
```

## Q15. The Hidden Chamber



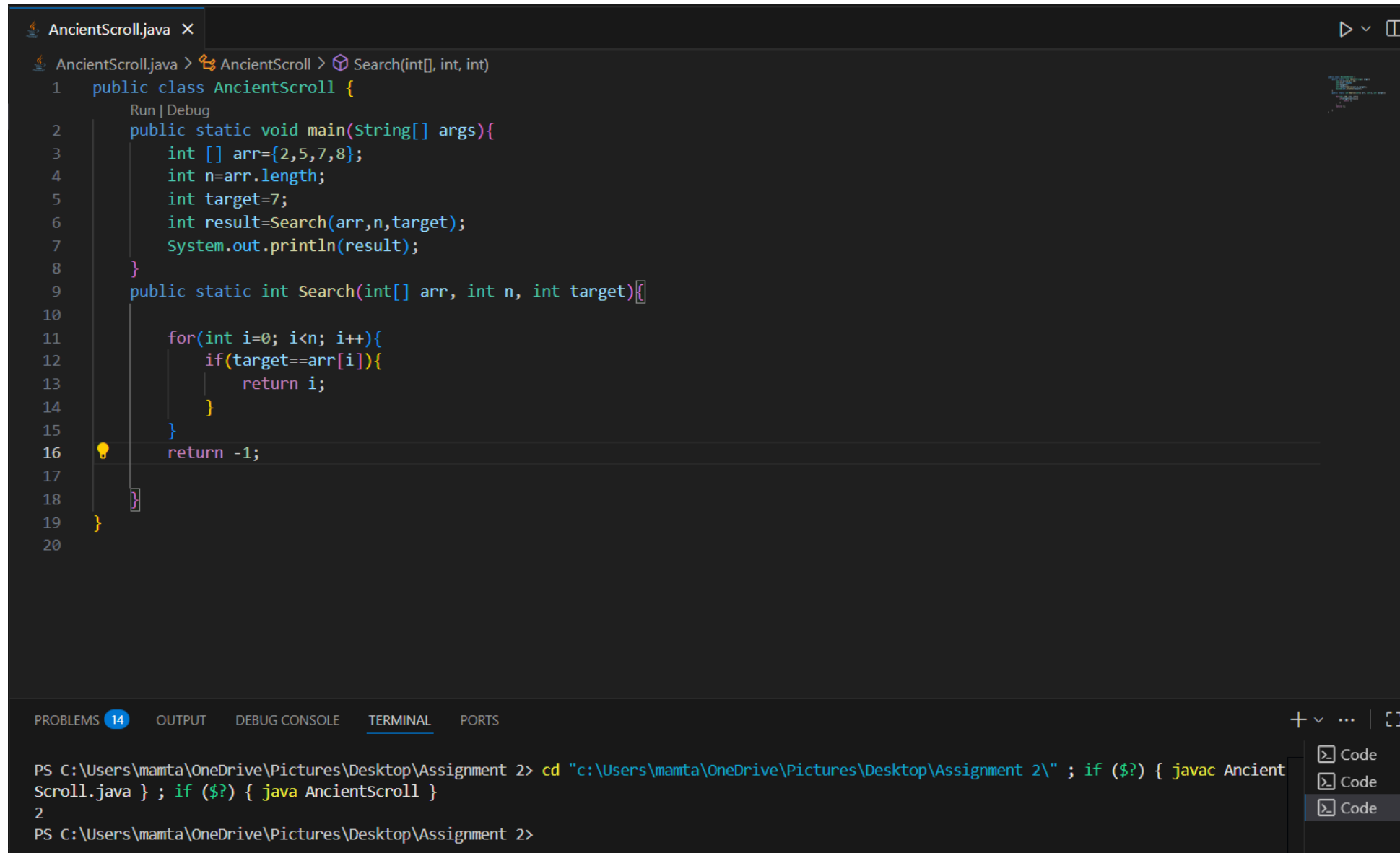
The screenshot shows an IDE with three tabs: ReverseString.java, Hidden.java (active), and Sum.java. The Hidden.java file contains the following code:

```
1 public class Hidden {  
2     public static void main(String[] args) {  
3         int arr[]={1,2,3,4};  
4         int n=arr.length;  
5         int result=sum(arr,n);  
6         System.out.println(result);  
7     }  
8     public static int sum(int arr[], int n){  
9         if (n == 0) {  
10             return 0;  
11         }  
12         return arr[n - 1] + sum(arr, n - 1);  
13     }  
14 }  
15
```

The bottom of the IDE shows the terminal output and command prompt. The command prompt shows the directory path and the command to compile and run the program:

```
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Hidden.java } ; java Hidden  
10  
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q16. The Ancient Scroll



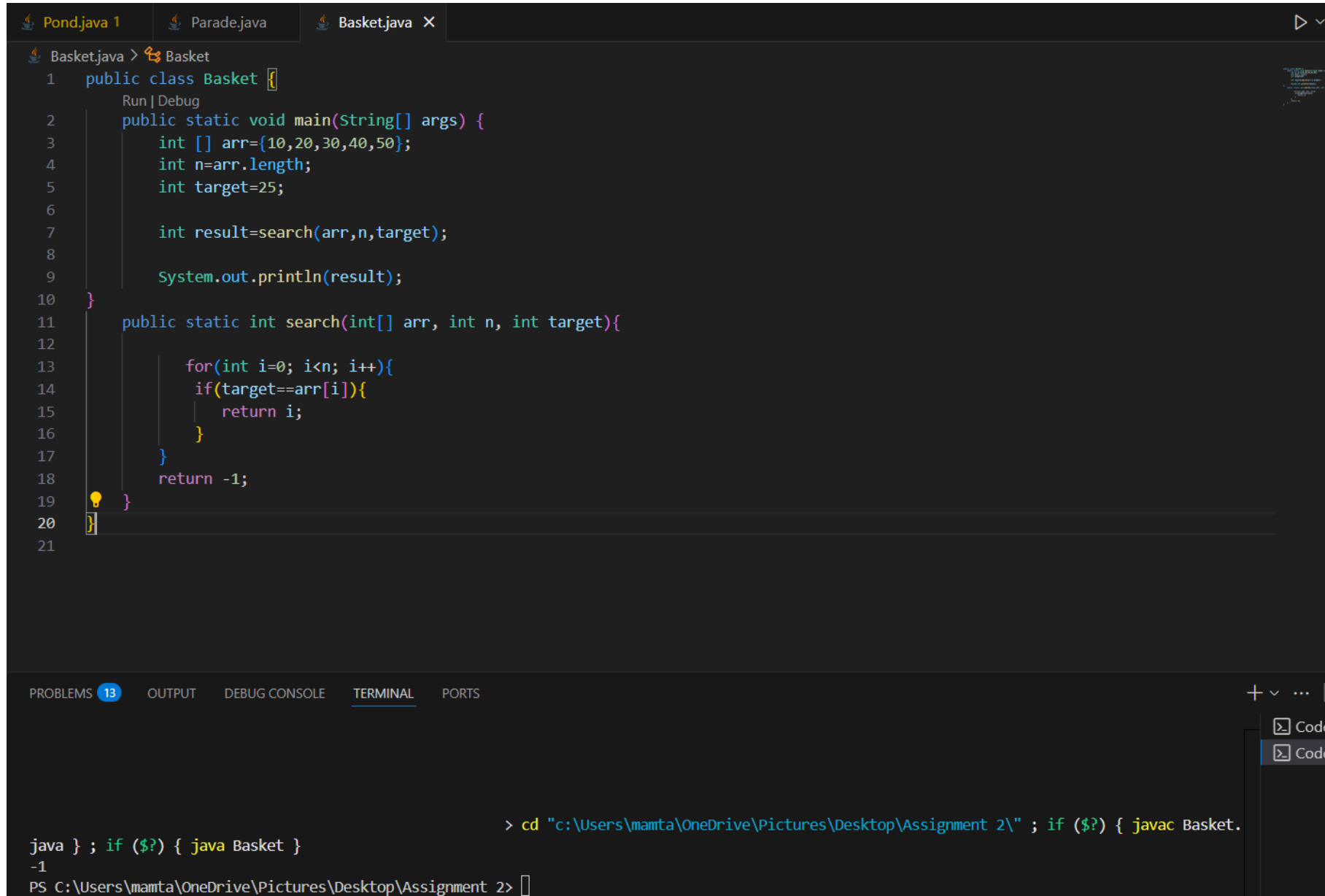
The screenshot shows an IDE with a file named `AncientScroll.java`. The code defines a class `AncientScroll` with a `main` method and a `Search` method. The `main` method initializes an array `arr` with values `{2, 5, 7, 8}`, sets `n` to the array length (4), sets `target` to 7, and calls `Search(arr, n, target)`. The `Search` method iterates through the array and returns the index of the first occurrence of the target, or -1 if not found. A lightbulb icon is present next to the `return -1;` statement on line 16. The bottom panel shows the terminal with the command to compile and run the program.

```
AncientScroll.java X
AncientScroll.java > AncientScroll > Search(int[], int, int)
1 public class AncientScroll {
2     Run | Debug
3     public static void main(String[] args){
4         int [] arr={2,5,7,8};
5         int n=arr.length;
6         int target=7;
7         int result=Search(arr,n,target);
8         System.out.println(result);
9     }
10    public static int Search(int[] arr, int n, int target){
11        for(int i=0; i<n; i++){
12            if(target==arr[i]){
13                return i;
14            }
15        }
16        return -1;
17    }
18 }
19 }
20 }

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Ancient
Scroll.java } ; if ($?) { java AncientScroll }
2
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```



## Q17. The Farmer's Basket

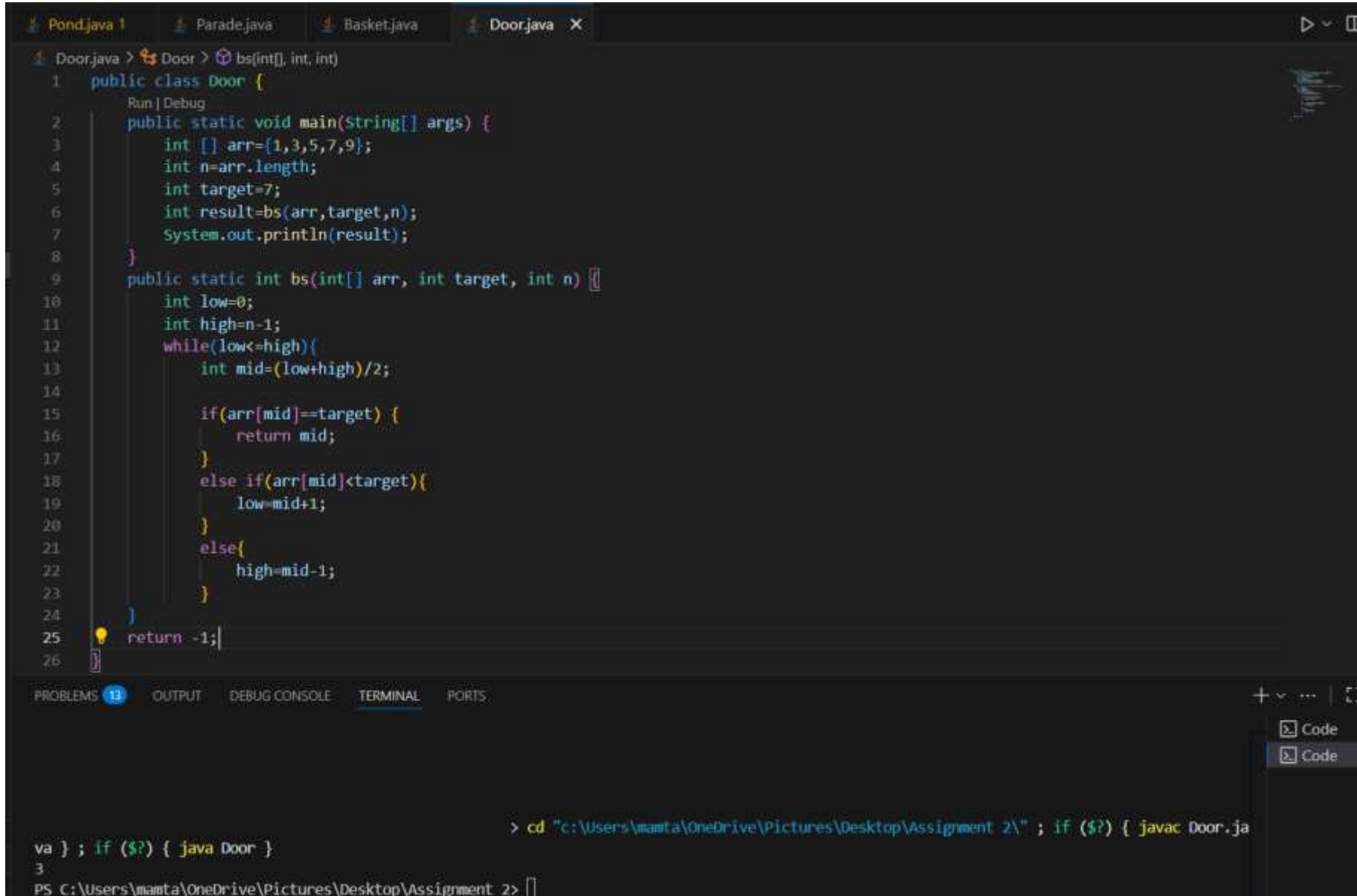


```
1 public class Basket {
2     public static void main(String[] args) {
3         int [] arr={10,20,30,40,50};
4         int n=arr.length;
5         int target=25;
6
7         int result=search(arr,n,target);
8
9         System.out.println(result);
10    }
11    public static int search(int[] arr, int n, int target){
12
13        for(int i=0; i<n; i++){
14            if(target==arr[i]){
15                return i;
16            }
17        }
18        return -1;
19    }
20 }
21
```

PROBLEMS 13 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Basket.
java } ; if ($?) { java Basket }
-1
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q18. The Secret Door

The image shows a screenshot of an IDE with a dark theme. At the top, there are tabs for 'Pond.java', 'Parade.java', 'Basket.java', and 'Door.java'. The 'Door.java' tab is active. The code in the editor is a Java program for binary search. It starts with a 'public class Door' and a 'main' method that initializes an array, sets a target, and calls a 'bs' method. The 'bs' method implements a binary search algorithm. The bottom of the screen shows a terminal window with a command prompt. The command prompt shows the current directory and a command to compile and run the program. The IDE interface includes a 'Run' button, a 'Debug' button, and a 'Terminal' tab. The 'Terminal' tab is active, showing the command prompt. The command prompt shows the current directory and a command to compile and run the program. The command is: `cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Door.java } ; if ($?) { java Door }`. The prompt is: `PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>`.

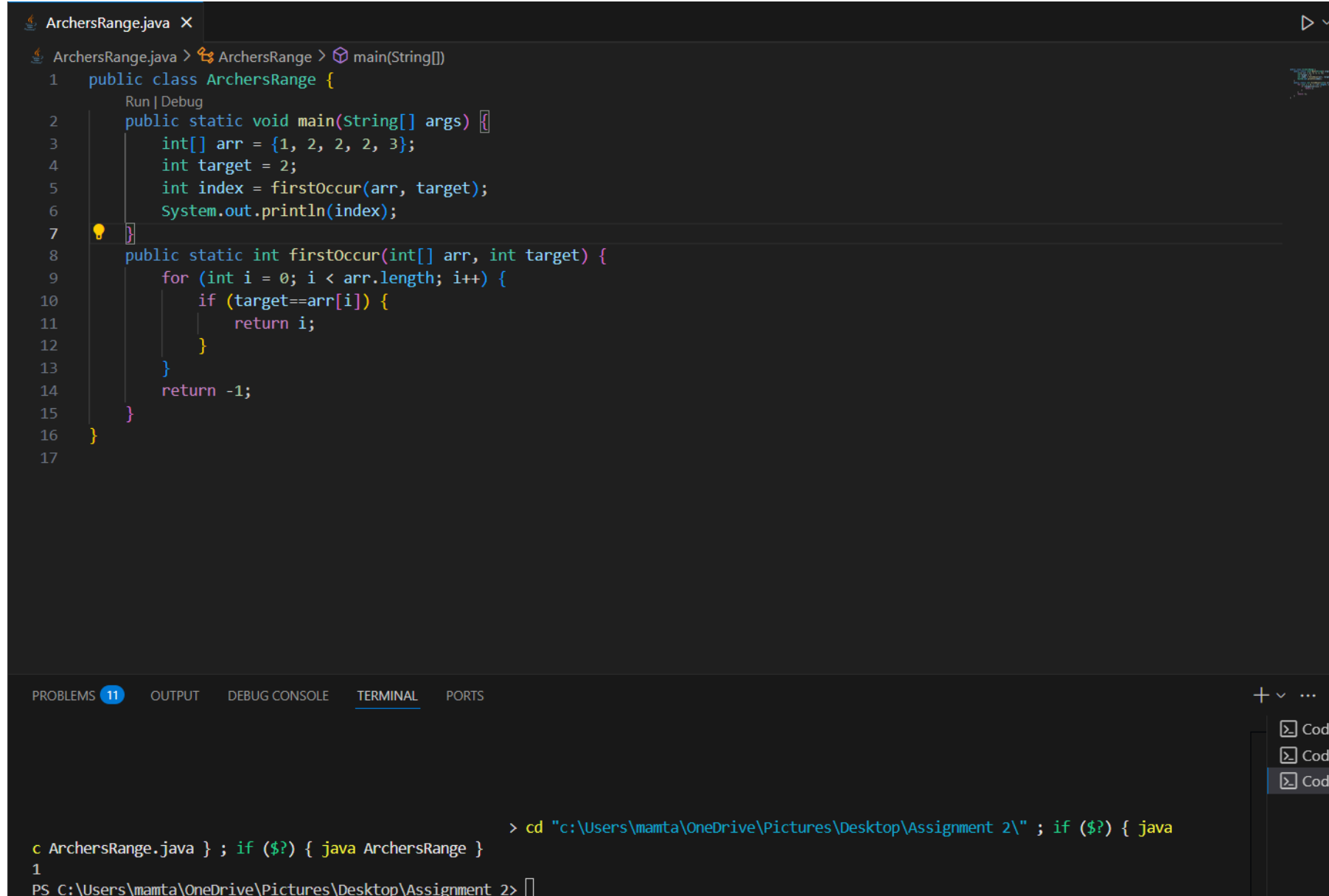
```
Door.java > Door > bs(int[], int, int)
1 public class Door {
2     public static void main(String[] args) {
3         int [] arr={1,3,5,7,9};
4         int n=arr.length;
5         int target=7;
6         int result=bs(arr,target,n);
7         System.out.println(result);
8     }
9     public static int bs(int[] arr, int target, int n) {
10        int low=0;
11        int high=n-1;
12        while(low<=high){
13            int mid=(low+high)/2;
14
15            if(arr[mid]==target) {
16                return mid;
17            }
18            else if(arr[mid]<target){
19                low=mid+1;
20            }
21            else{
22                high=mid-1;
23            }
24        }
25        return -1;
26    }
}
```

PROBLEMS 13 OUTPUT DEBUG CONSOLE TERMINAL PORTS

> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if (\$?) { javac Door.java } ; if (\$?) { java Door }

PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>

## Q19. The Archer's Range



The screenshot shows an IDE with a file named `ArchersRange.java`. The code defines a class `ArchersRange` with a `main` method and a `firstOccur` method. The `main` method initializes an array `arr = {1, 2, 2, 2, 3}`, sets `target = 2`, and calls `firstOccur(arr, target)` to find the first index of the target. The `firstOccur` method iterates through the array and returns the index of the first occurrence of the target, or `-1` if not found. The IDE interface includes a terminal at the bottom with a command prompt showing the execution of the program.

```
ArchersRange.java X
ArchersRange.java > ArchersRange > main(String[])
1 public class ArchersRange {
2     public static void main(String[] args) {
3         int[] arr = {1, 2, 2, 2, 3};
4         int target = 2;
5         int index = firstOccur(arr, target);
6         System.out.println(index);
7     }
8     public static int firstOccur(int[] arr, int target) {
9         for (int i = 0; i < arr.length; i++) {
10             if (target == arr[i]) {
11                 return i;
12             }
13         }
14         return -1;
15     }
16 }
17

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { java
c ArchersRange.java } ; if ($?) { java ArchersRange }
1
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q20. The Treasure Chest

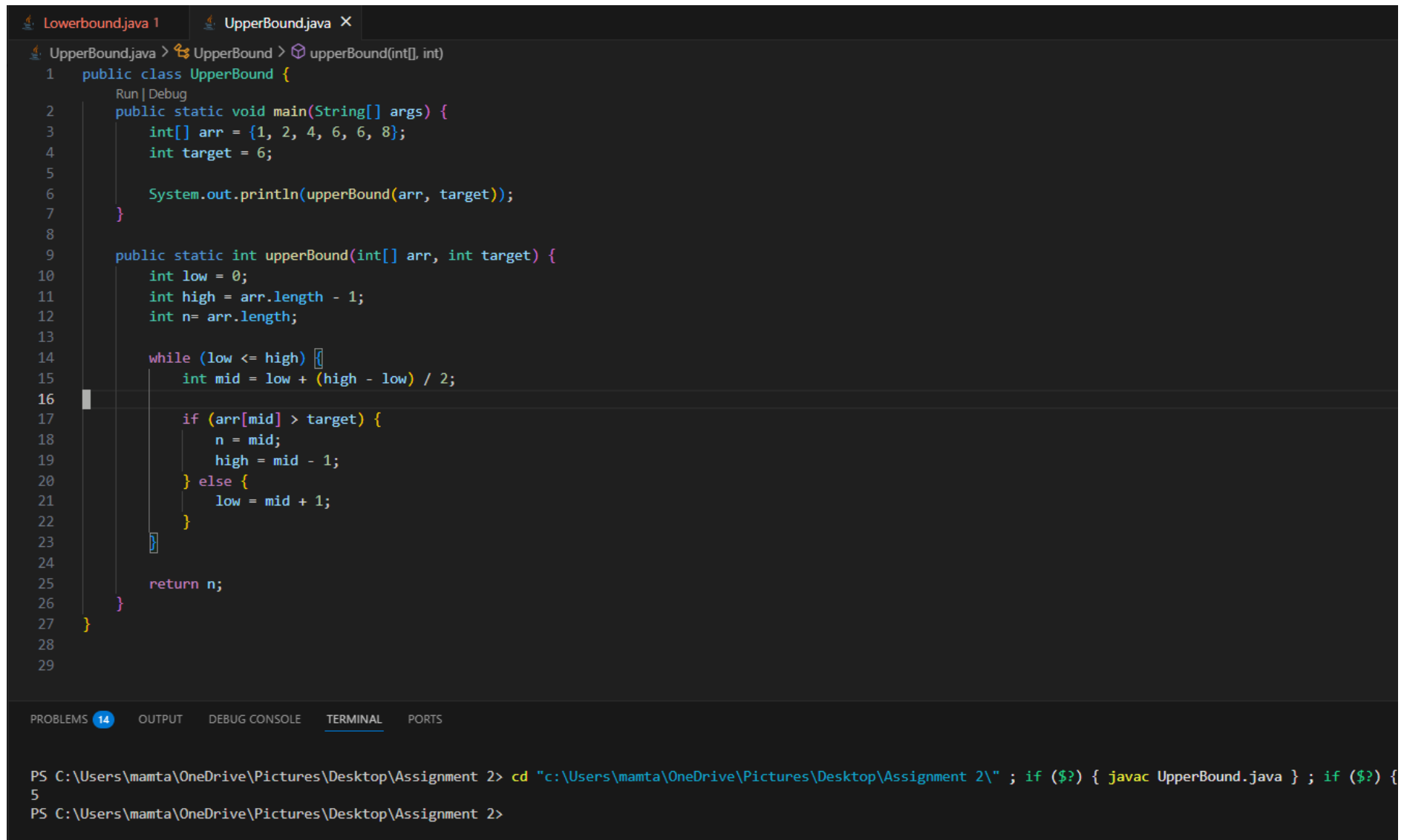
```
TreasureChest.java X
TreasureChest.java > TreasureChest > lastOccurrence(int[], int)
1 public class TreasureChest {
2     Run | Debug
3     public static void main(String[] args) {
4         int[] arr = {1, 2, 2, 2, 3};
5         int target = 2;
6         int index = lastOccurrence(arr, target);
7         System.out.println(index);
8     }
9
10    public static int lastOccurrence(int[] arr, int target) {
11        int low = 0;
12        int high = arr.length - 1;
13        int result = -1;
14        while (low <= high) {
15            int mid = low + (high - low) / 2;
16            if (arr[mid] == target) {
17                result = mid;
18                low = mid + 1;
19            } else if (arr[mid] < target) {
20                low = mid + 1;
21            } else {
22                high = mid - 1;
23            }
24        }
25        return result;
26    }
27 }

PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac TreasureChest.java } ; if ($?) { java TreasureChest }
3
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

Q21.The first index where the element is greater than or equal to the target.

```
Lowerbound.java x
Lowerbound.java > Lowerbound
1 public class Lowerbound {
    Run [Debug]
2     public static void main(String[] args) {
3         int[] arr = {1, 2, 4, 6, 6, 8};
4         int target = 6;
5         System.out.println(lowerBound(arr, target));
6     }
7
8     public static int lowerBound(int[] arr, int target) {
9         int low = 0;
10        int high = arr.length - 1;
11        int n = arr.length;
12
13        while (low <= high) {
14            int mid = low + (high - low) / 2;
15
16            if (arr[mid] >= target) {
17                n = mid;
18                high = mid - 1;
19            } else {
20                low = mid + 1;
21            }
22        }
23    }
24 }
25
PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> c
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Lowerbound.java } ; if ($?) { ja
rbound }
3
```

Q22.The first index where the element is strictly greater than the target.



```
LowerBound.java 1  UpperBound.java X
UpperBound.java > UpperBound > upperBound(int[], int)
1  public class UpperBound {
    Run | Debug
2  public static void main(String[] args) {
3      int[] arr = {1, 2, 4, 6, 6, 8};
4      int target = 6;
5
6      System.out.println(upperBound(arr, target));
7  }
8
9  public static int upperBound(int[] arr, int target) {
10     int low = 0;
11     int high = arr.length - 1;
12     int n = arr.length;
13
14     while (low <= high) {
15         int mid = low + (high - low) / 2;
16
17         if (arr[mid] > target) {
18             n = mid;
19             high = mid - 1;
20         } else {
21             low = mid + 1;
22         }
23     }
24
25     return n;
26 }
27
28
29

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac UpperBound.java } ; if ($?) {
5
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q23.The smallest element $\geq$ target (actual value, not index).

```
Lowerbound.java 1 UpperBound.java Ceil.java X
Ceil.java > Ceil > findCeil(int[], int)
1 public class Ceil {
  Run | Debug
2 public static void main(String[] args) {
3     int[] arr = {1, 2, 4, 6, 6, 8};
4     int target = 5;
5
6     System.out.println("Ceil Element: " + findCeil(arr, target));
7 }
8
9 public static int findCeil(int[] arr, int target) {
10     int low = 0, high = arr.length - 1;
11     int ans = -1;
12
13     while (low <= high) {
14         int mid = low + (high - low) / 2;
15
16         if (arr[mid] >= target) {
17             ans = arr[mid];
18             high = mid - 1;
19         } else {
20             low = mid + 1;
21         }
22     }
23
24     return ans;
25 }
26 }
27

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Ceil.java } ; if ($?) { java Ceil }
Ceil Element: 6
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q24. The largest element $\leq$ target.

```
Floor.java X
Floor.java > Floor > findFloor(int[], int)
1 public class Floor{
    Run | Debug
2     public static void main(String[] args) {
3         int[] arr = {1, 2, 4, 6, 6, 8};
4         int target = 5;
5
6         System.out.println("Floor Element: " + findFloor(arr, target));
7     }
8
9     public static int findFloor(int[] arr, int target) {
10        int low = 0, high = arr.length - 1;
11        int ans = -1;
12
13        while (low <= high) {
14            int mid = (low+high) / 2;
15
16            if (arr[mid] <= target) {
17                ans = arr[mid];
18                low = mid + 1;
19            } else {
20                high = mid - 1;
21            }
22        }
23
24        return ans;
25    }
26 }
27
28
```

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> c
> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Floor.java } ; if ($?) { java Floor }
Floor Element: 4
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```



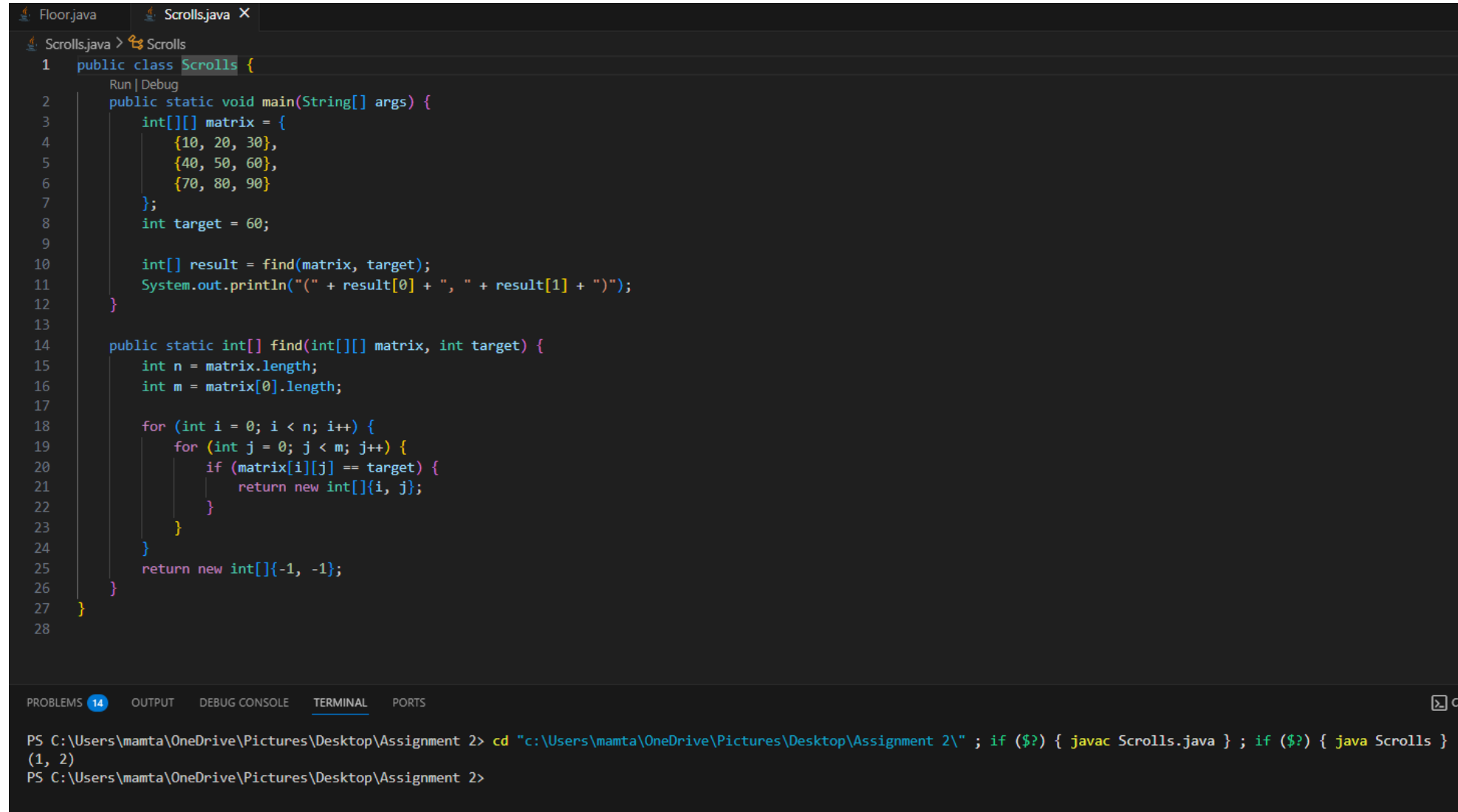
## Q25. The Treasure Map (Linear Search)

```
TreasureMap.java X
TreasureMap.java > TreasureMap > findTreasure(int[][], int)
1 public class TreasureMap {
2     Run | Debug
3     public static void main(String[] args) {
4         int[][] matrix = {
5             {1, 2, 3},
6             {4, 5, 6},
7             {7, 8, 9}
8         };
9         int target = 5;
10        if (findTreasure(matrix, target)) {
11            System.out.println(x:"Yes");
12        } else {
13            System.out.println(x:"No");
14        }
15    }
16
17    public static boolean findTreasure(int[][] matrix, int target) {
18        int n = matrix.length;
19        int m = matrix[0].length;
20        for (int i = 0; i < n; i++) {
21            for (int j = 0; j < m; j++) {
22                if (matrix[i][j] == target) {
23                    return true;
24                }
25            }
26        }
27        return false;
28    }
29 }
30
```

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac TreasureMap.java } ; if ($?) {
Yes
```

## Q26. The Magical Scrolls (Linear Search Return Index)



```
Scrolls.java > Scrolls
1 public class Scrolls {
2     public static void main(String[] args) {
3         int[][] matrix = {
4             {10, 20, 30},
5             {40, 50, 60},
6             {70, 80, 90}
7         };
8         int target = 60;
9
10        int[] result = find(matrix, target);
11        System.out.println("(" + result[0] + ", " + result[1] + ")");
12    }
13
14    public static int[] find(int[][] matrix, int target) {
15        int n = matrix.length;
16        int m = matrix[0].length;
17
18        for (int i = 0; i < n; i++) {
19            for (int j = 0; j < m; j++) {
20                if (matrix[i][j] == target) {
21                    return new int[]{i, j};
22                }
23            }
24        }
25        return new int[]{-1, -1};
26    }
27 }
28
```

PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2> cd "c:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2\" ; if ($?) { javac Scrolls.java } ; if ($?) { java Scrolls }
(1, 2)
PS C:\Users\mamta\OneDrive\Pictures\Desktop\Assignment 2>
```

## Q30. The Magic Portal (Binary Search 2D)

```
TreasureMap.java MagicPortal.java X
MagicPortal.java > MagicPortal > activatePortal(int[][], int)
1 public class MagicPortal {
2     Run | Debug
3     public static void main(String[] args) {
4         int[][] matrix = {
5             {1, 2, 8},
6             {3, 6, 10},
7             {7, 9, 12}
8         };
9         int target = 9;
10
11         if (activatePortal(matrix, target)) {
12             System.out.println(x: "Activated");
13         } else {
14             System.out.println(x: "Failed");
15         }
16     }
17
18     public static boolean activatePortal(int[][] matrix, int target) {
19         int n = matrix.length;
20         int m = matrix[0].length;
21         int row = 0, col = m - 1;
22
23         while (row < n && col >= 0) {
24             if (matrix[row][col] == target) {
25                 return true;
26             } else if (matrix[row][col] > target) {
27                 col--;
28             } else {
29                 row++;
30             }
31         }
32         return false;
33     }
34 }
35
```