DSA PRACTICE SET 3

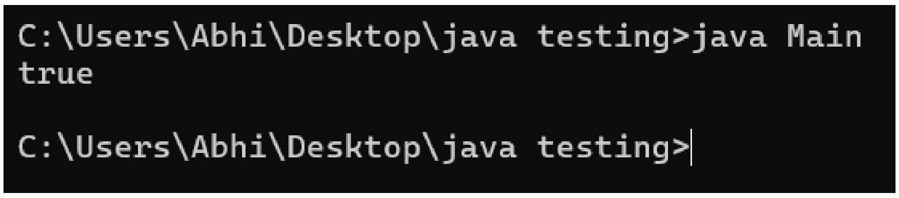
B Abhimanyu 22CB006

1) Anagram Program

CODE

```
import java.util.Arrays;
public class Main {
   public static void main(String[] args) {
      String s1 = "Geeks";
      String s2 = "skeeG";
      if(s1.length()!=s2.length())System.out.println(false);;
      char[] arr1 = s1.toCharArray();
      char[] arr2 = s2.toCharArray();
      Arrays.sort(arr1);
      Arrays.sort(arr2);
      s1 = new String(arr1);
      s2 = new String(arr2);
      System.out.println(s1.equals(s2));
   }}
```

OUTPUT



```
C:\Users\Abhi\Desktop\java testing>java Main
true

C:\Users\Abhi\Desktop\java testing>
```

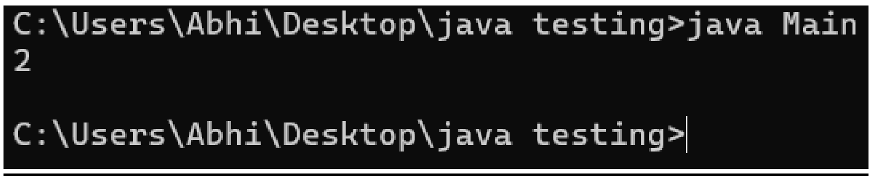
Time Complexity: O(nlogn)
Space Complexity: O(n)

2) Rows with Max 1s

CODE

```
import java.util.Arrays;
public class Main {
   public static void main(String[] args) {
      int[][] arr = {{0, 1, 1, 1},
                {0, 0, 1, 1},
                {1, 1, 1, 1},
                {0, 0, 0, 0}};
      int maxRowIndex = -1;
      int maxOnes = 0;
      for (int i = 0; i < arr.length; i++) {
            int ones = 0;
            for (int j = 0; j < arr[0].length; j++) {
                if (arr[i][j] == 1) {
                    ones++;
                }
            }
            if (ones > maxOnes) {
                maxOnes = ones;
                maxRowIndex = i;
            }
        }
        System.out.println(maxRowIndex);
    }
}
```

OUTPUT



```
C:\Users\Abhi\Desktop\java testing>java Main
2

C:\Users\Abhi\Desktop\java testing>
```

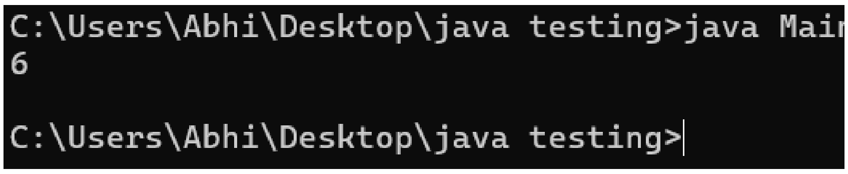
Time Complexity: O(n*m)
Space Complexity: O(1)

3) Longest Consecutive Subsequence

CODE

```
import java.util.Arrays;
public class Main {
    public static void main(String[] args) {
        int[] arr = {2, 6, 1, 9, 4, 5, 3};
        Arrays.sort(arr);
        int longest = 1;
        int curr = 1;
        for(int i = 1; i< arr.length; i++){
            if (arr[i] == arr[i - 1]) {
                continue;
            } else if (arr[i] - arr[i - 1] == 1) {
                curr+=1;
            } else {
                longest = Math.max(longest, curr);
                curr = 1;
            }
        }
        longest = Math.max(curr, longest);
        System.out.println(longest);
    }
}
```

OUTPUT



```
C:\Users\Abhi\Desktop\java testing>java Mai
6

C:\Users\Abhi\Desktop\java testing>
```

Time Complexity: O(nlogn)

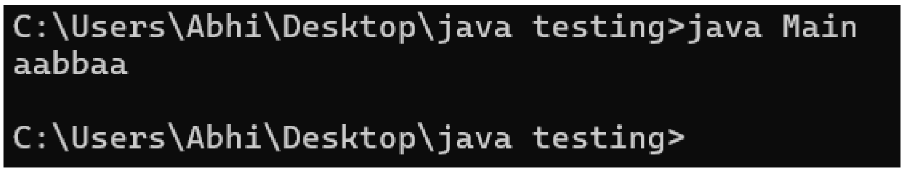
Space Complexity: O(1)

4) Longest Pallindrome in a String

CODE

```
import java.util.Arrays;
public class Main {
    public static void main(String[] args) {
        String s = "aaaabbaa";
        if (s == null || s.length() == 0) System.out.println("");
        int ans = 0;
        String longest = "";
        for (int i = 0; i < s.length(); i++) {
            int len = 1;
            int left = i - 1;
            int right = i + 1;
            while (left >= 0 && right < s.length() && s.charAt(left) == s.charAt(right)) {
                len += 2;
                left--;
                right++;
            }
            if (len > ans) {
                ans = len;
                longest = s.substring(left + 1, right);
            }
            len = 0;
            left = i;
            right = i + 1;
            while (left >= 0 && right < s.length() && s.charAt(left) == s.charAt(right)) {
                len += 2;
                left--;
                right++;
            }
            if (len > ans) {
                ans = len;
                longest = s.substring(left + 1, right);
            }
        }
        System.out.println(longest);
    }
}
```

OUTPUT



```
C:\Users\Abhi\Desktop\java testing>java Main
aabbaa

C:\Users\Abhi\Desktop\java testing>
```

Time Complexity: O(n^2)
Space Complexity: O(n)

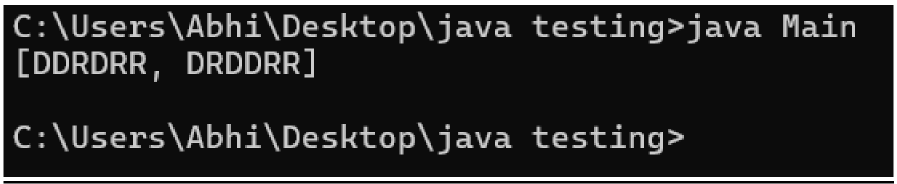
5) Rat in Maze

CODE

```
import java.util.ArrayList;
import java.util.Arrays;
public class Main {
    static ArrayList<String> res= new ArrayList<>();
    static void run(int[][] mat, int r, int c, int maxr, int maxc, String path){
        if(r<0 || r>maxr || c<0 || c>maxc || mat[r][c]==0){
            return;
        }
        if(r==maxr && c==maxc){
            res.add(path);
            return;
        }
        mat[r][c]=0;
        run(mat, r+1, c, maxr, maxc, path+"D");
        run(mat, r-1, c, maxr, maxc, path+"U");
        run(mat, r, c+1, maxr, maxc, path+"R");
        run(mat, r, c-1, maxr, maxc, path+"L");
        mat[r][c] = 1;
    }
    public static void main(String[] args) {
        int[][] mat ={{1, 0, 0, 0},
                    {1, 1, 0, 1},
                    {1, 1, 0, 0},
                    {0, 1, 1, 1}};

        if (mat[0][0] == 0 || mat[mat.length-1][mat[0].length-1] == 0) {
            System.out.println("");
        }
        run(mat, 0,0,mat.length-1, mat[0].length-1, "");
        System.out.println(res.toString());
    }
}
```

OUTPUT



```
C:\Users\Abhi\Desktop\java testing>java Main
[DDRDRR, DRDDRR]

C:\Users\Abhi\Desktop\java testing>
```

Time Complexity: O(4m×n)

Space Complexity: O(m×n)