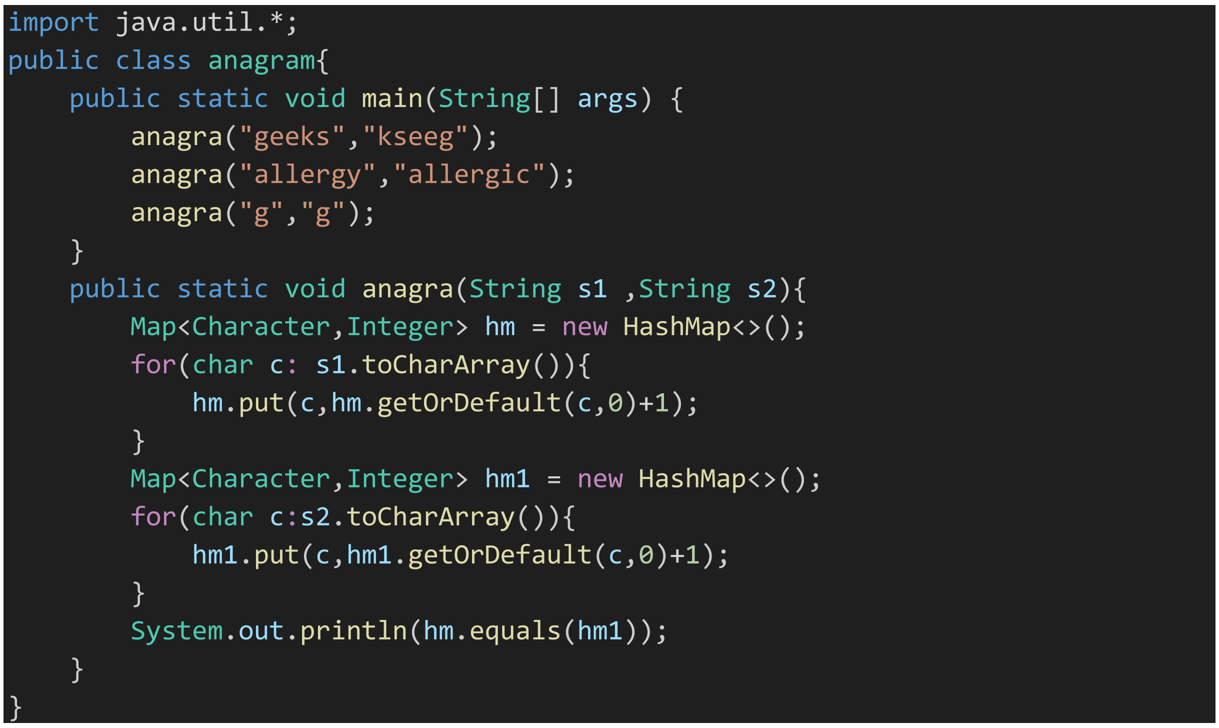
DSA PRACTICE – 3.

Name:Deepak S

Register No: 22IT018

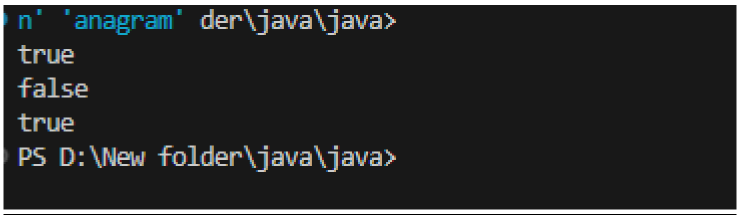
<u>Anagram program</u>

<u>CODE :</u>



```java
import java.util.*;
public class anagram{
    public static void main(String[] args) {
        anagra("geeks","kseeg");
        anagra("allergy","allergic");
        anagra("g","g");
    }
    public static void anagra(String s1 ,String s2){
        Map<Character,Integer> hm = new HashMap<>();
        for(char c: s1.toCharArray()){
            hm.put(c,hm.getOrDefault(c,0)+1);
        }
        Map<Character,Integer> hm1 = new HashMap<>();
        for(char c:s2.toCharArray()){
            hm1.put(c,hm1.getOrDefault(c,0)+1);
        }
        System.out.println(hm.equals(hm1));
    }
}
```

<u>OUTPUT :</u>



```
n' 'anagram' der\java\java>
true
false
true
PS D:\New folder\java\java>
```

TIME COMPLEXITY : O(n)

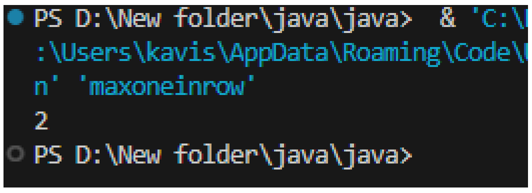
SPACE COMPLEXITY : O(n)

row with max 1s'

CODE :

```java
public class maxoneinrow {
    public static void main(String[] args) {
        System.out.println(find(new int[][]{{0, 1, 1, 1}, {0, 0, 1, 1}, {1, 1, 
1, 1}, {0, 0, 0, 0}}));
    }
    public static int find(int arr[][]) {
        int key = -1 , value = 0;
        for(int i = 0 ; i<arr.length; i++){
            int one = 0;
            for(int j =0 ; j<arr[i].length ; j++){
                if(arr[i][j]==1) one++;
            }
            if(value<one){
                key = i;
                value = one;
            }
        }
        return key;
    }
}
```

OUTPUT :



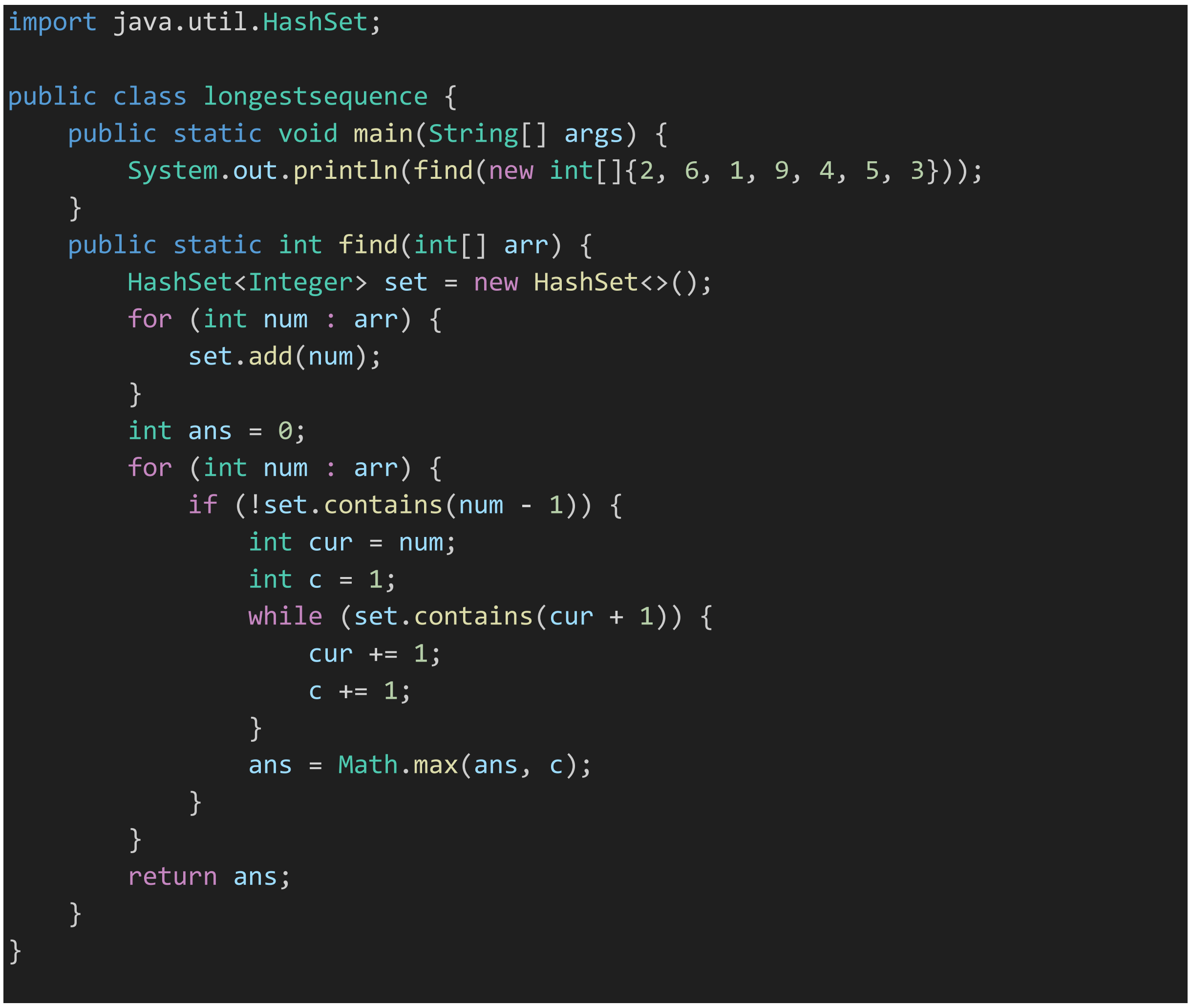
```
PS D:\New folder\java\java>  & 'C:\
:\Users\kavis\AppData\Roaming\Code\
n' 'maxoneinrow'
2
PS D:\New folder\java\java>
```

TIME COMPLEXITY : O(m*n)

SPACE COMPLEXITY : O(1)

Longest consequtive subsequence

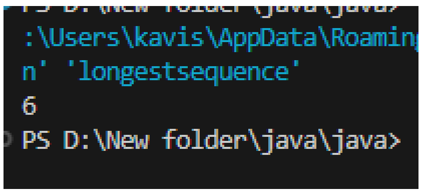
CODE :



```java
import java.util.HashSet;

public class longestsequence {
    public static void main(String[] args) {
        System.out.println(find(new int[]{2, 6, 1, 9, 4, 5, 3}));
    }
    public static int find(int[] arr) {
        HashSet<Integer> set = new HashSet<>();
        for (int num : arr) {
            set.add(num);
        }
        int ans = 0;
        for (int num : arr) {
            if (!set.contains(num - 1)) {
                int cur = num;
                int c = 1;
                while (set.contains(cur + 1)) {
                    cur += 1;
                    c += 1;
                }
                ans = Math.max(ans, c);
            }
        }
        return ans;
    }
}
```

OUTPUT:



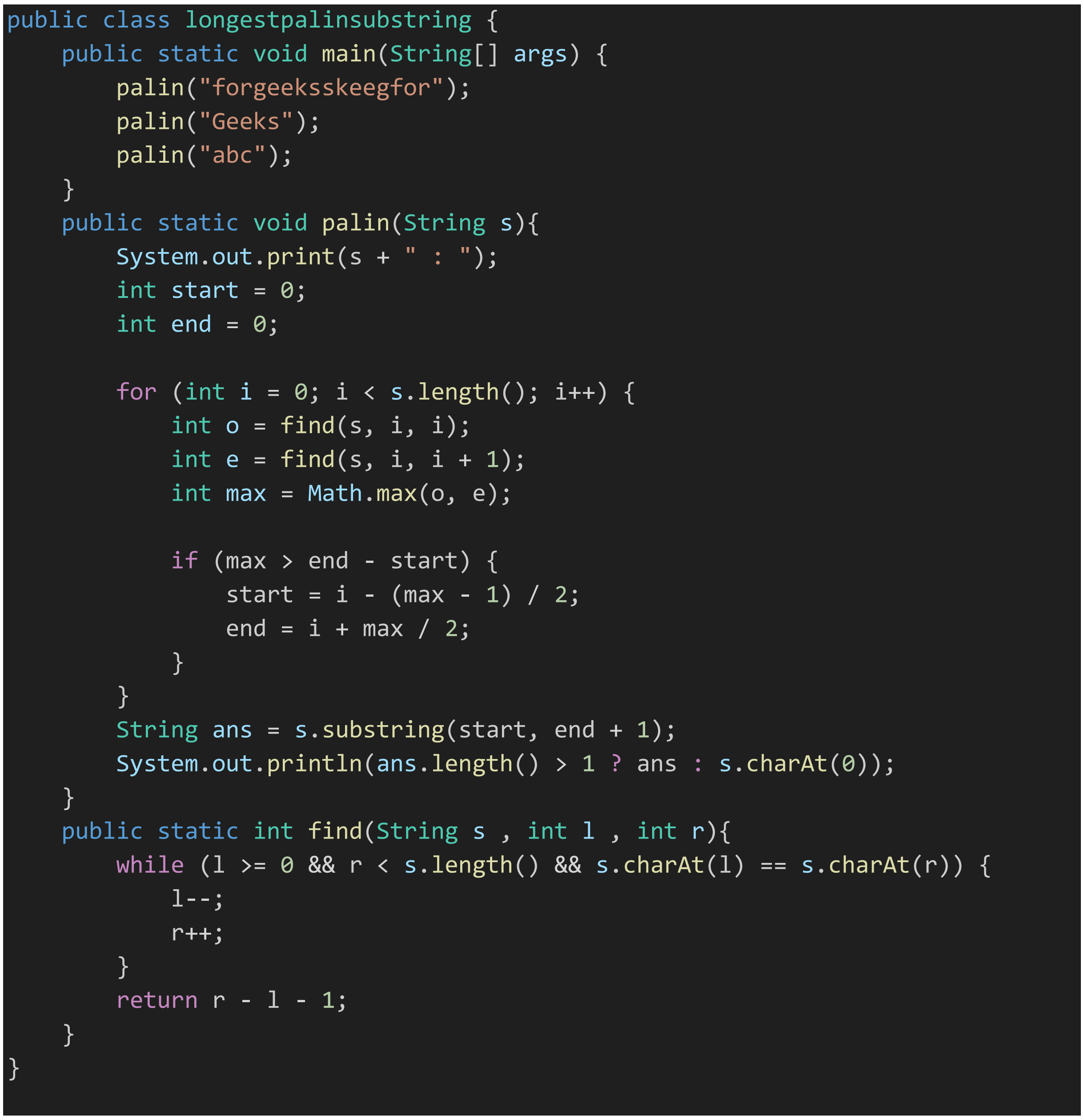
```
PS D:\New folder\java\java>
:\Users\kavis\AppData\Roamin
n' 'longestsequence'
6
PS D:\New folder\java\java>
```

TIME COMPLEXITY : O(n)

SPACE COMPLEXITY : O(n)

longest palindrome in a string

CODE :

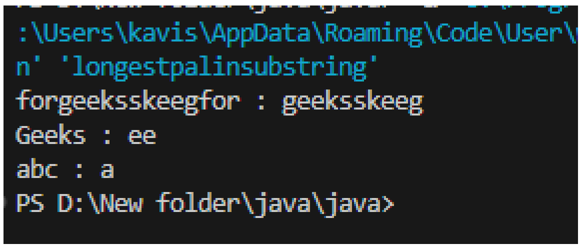


```java
public class longestpalinsubstring {
    public static void main(String[] args) {
        palin("forgeeksskeegfor");
        palin("Geeks");
        palin("abc");
    }
    public static void palin(String s){
        System.out.print(s + " : ");
        int start = 0;
        int end = 0;

        for (int i = 0; i < s.length(); i++) {
            int o = find(s, i, i);
            int e = find(s, i, i + 1);
            int max = Math.max(o, e);

            if (max > end - start) {
                start = i - (max - 1) / 2;
                end = i + max / 2;
            }
        }
        String ans = s.substring(start, end + 1);
        System.out.println(ans.length() > 1 ? ans : s.charAt(0));
    }
    public static int find(String s , int l , int r){
        while (l >= 0 && r < s.length() && s.charAt(l) == s.charAt(r)) {
            l--;
            r++;
        }
        return r - l - 1;
    }
}
```

OUTPUT:



```
[illegible]
:\Users\kavis\AppData\Roaming\Code\User\
n' 'longestpalinsubstring'
forgeeksskeegfor : geeksskeeg
Geeks : ee
abc : a
PS D:\New folder\java\java>
```

TIME COMPLEXITY : O(n^2)

SPACE COMPLEXITY : O(n)

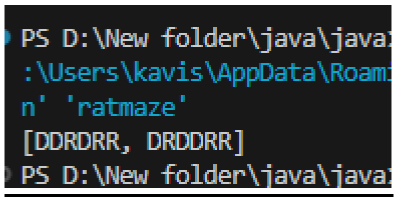
rat in a maze problem

CODE :

```java
import java.util.ArrayList;
public class ratmaze {
    public static void main(String[] args) {
        System.out.println(findPath(new int[][]{{1, 0, 0, 0}, {1, 1, 0, 1},
{1, 1, 0, 0}, {0, 1, 1, 1}}));
    }
    public static ArrayList<String> findPath(int[][] mat) {
        ArrayList<String> ans = new ArrayList<>();
        int n = mat.length;
        boolean[][] vis = new boolean[n][n];
        find(ans, 0, 0, mat, n, "", vis);
        return ans;
    }
    public static void find(ArrayList<String> ans, int r, int c, int[][] m,
int n, String p, boolean[][] vis) {
        if (r == n - 1 && c == n - 1 && m[r][c] != 0) {
            ans.add(p);
            return;
        }
        if (r >= 0 && r < n && c >= 0 && c < n) {
            if (vis[r][c] || m[r][c] == 0) {
                return;
            }
            vis[r][c] = true;
            find(ans, r + 1, c, m, n, p + 'D', vis);
            find(ans, r, c - 1, m, n, p +'L', vis);
            find(ans, r, c + 1, m, n, p + 'R', vis);
            find(ans, r - 1, c, m, n, p +'U', vis);
            vis[r][c] = false;
        }
    }

}
```

OUTPUT :



```
PS D:\New folder\java\java
:\Users\kavis\AppData\Roam
n' 'ratmaze'
[DDRDRR, DRDDRR]
PS D:\New folder\java\java
```

TIME COMPLEXITY : O(4^(n^2))

SPACE COMPLEXITY : O(n^2)