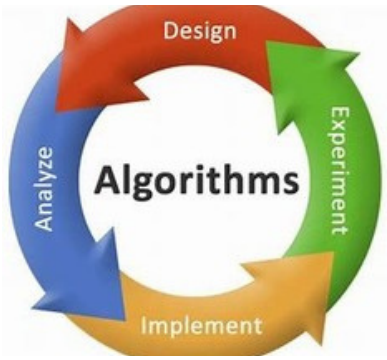
UPES



DAA LAB-3 Report File

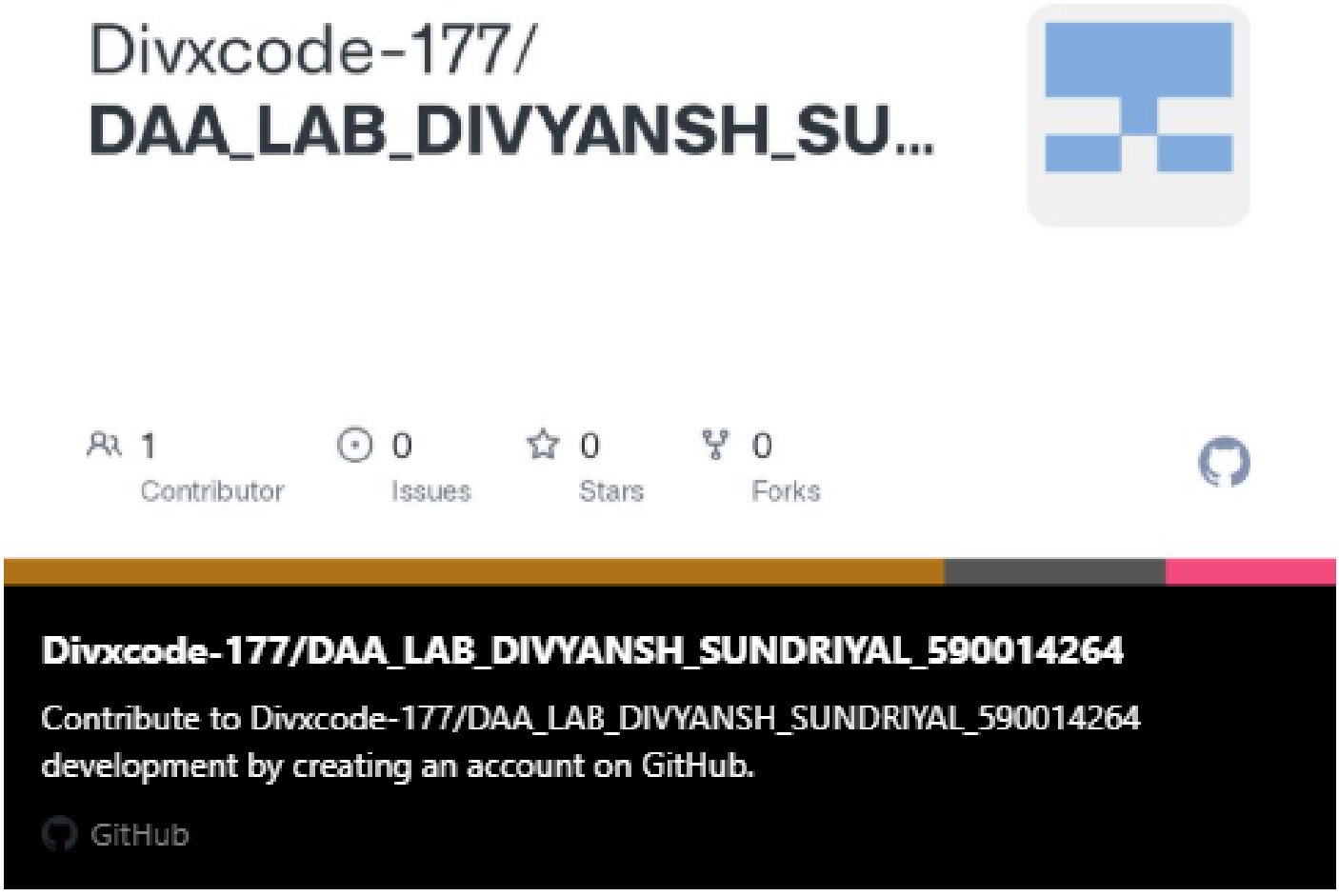
Name: Divyansh Sundriyal

Batch : B-33

SAP ID : 500064

REPOSITRY

https://github.com/Divxcode-177/DAA_LAB_DIVYANSH_SUNDRIYAL_590014264



LAB-3

Lab Experiment 3 – Implement the celebrity problem and analyze its time complexity.

Objective:

Implement the celebrity problem and analyze its time complexity.
programming language (C, C++, Java).
Also attach plagiarism report in the end.

CODE :

```
import java.util.Scanner;
import java.util.Stack;
public class celebrityproblem1{
  public static int celebrityProblemFunction(int[][] arr, int n) {
    Stack<Integer> stack = new Stack<>();
    for (int i = 0; i < n; i++) {
      stack.push(i);
    }
    while (stack.size() > 1) {
      int a = stack.pop();
      int b = stack.pop();
      if (arr[a][b] == 1) {
        stack.push(b);
      } else {
        stack.push(a);
      }
    }

    int candidate = stack.pop();
    for (int i = 0; i < n; i++) {
    if (i != candidate && (arr[candidate][i] == 1 || arr[i][candidate] == 0)) {
        return -1;
      }
    }
    return candidate;
  }
```

CODE :

```
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
final int n = 2;
int[][] arr = new int[n][n];
System.out.println("Enter the 2x2 matrix:");
for (int i = 0; i < n; i++) {
for (int j = 0; j < n; j++) {
arr[i][j] = sc.nextInt();
}
}
int celeb = celebrityProblemFunction(arr, n);
if (celeb == -1) {
System.out.println("No celebrity found");
} else {
System.out.println("Celebrity is person " + celeb);
}
sc.close();
}
}
```

OUTPUT:

```
PS C:\Users\dell\Documents\GitHub\DAA_LAB_DIVYANSH_SUNDRIYAL_590014264\LAB3\CODE> java celebrityproblem1.java
Enter the 2x2 matrix:
1 0
0 1
PS C:\Users\dell\Documents\GitHub\DAA_LAB_DIVYANSH_SUNDRIYAL_590014264\LAB3\CODE> java celebrityproblem1.java
Enter the 2x2 matrix:
1 1
1 1
No celebrity found
PS C:\Users\dell\Documents\GitHub\DAA_LAB_DIVYANSH_SUNDRIYAL_590014264\LAB3\CODE> java celebrityproblem1.java
Enter the 2x2 matrix:
1 2
3 4
Celebrity is person 1
PS C:\Users\dell\Documents\GitHub\DAA_LAB_DIVYANSH_SUNDRIYAL_590014264\LAB3\CODE>
```

LAB-3

Analyze time complexity of Celebrity Problem

Time Complexity Analysis

- **Step 1 (Candidate finding): Each iteration eliminates one person → O(n).**
- **Step 2 (Verification): Check all other n-1 people → O(n).**

Total = O(n) time, O(1) space

Celebrity algorithm works in two phases:

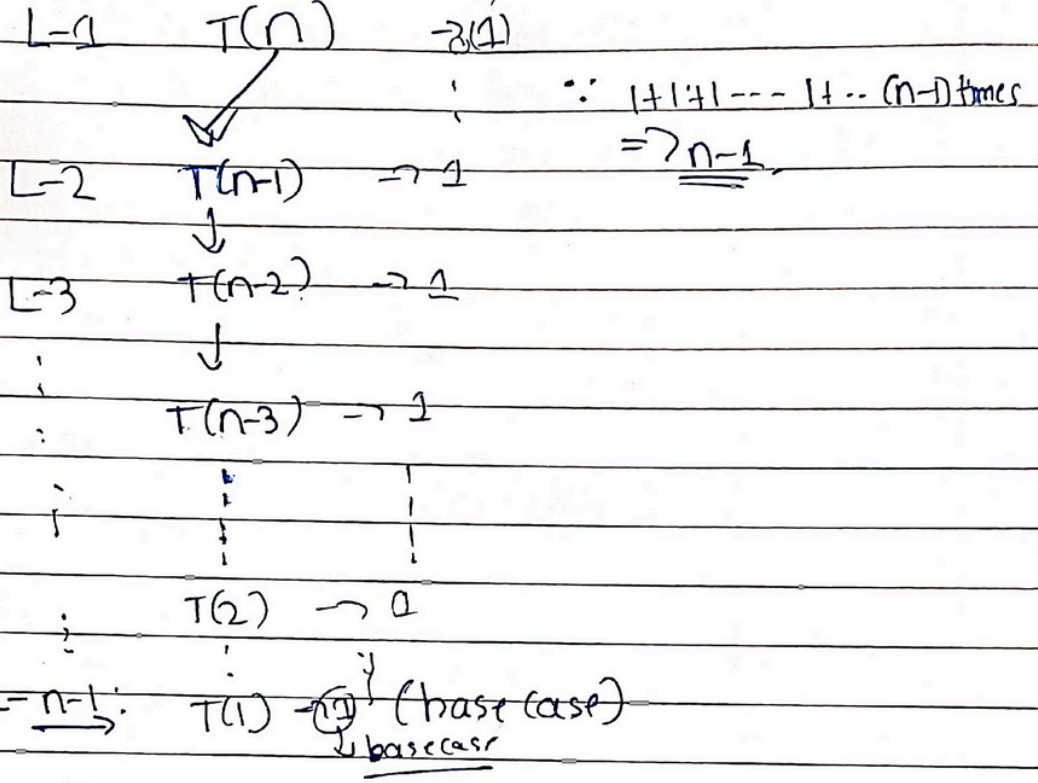
1. **Candidate Selection (using stack):**
   - **Push all n people in stack → O(n)**
   - **While stack size > 1: Pop two, push one back → eliminates one person each iteration → n-1 comparisons → O(n)**
2. **Candidate Verification:**
   - **Check candidate against all n-1 people → O(n)**

Total iterative implementation: O(n + n) = O(n)

PAGE NO.:
DATE: / /

Analyse of Celebrity problem Time complexity

(1) ∵ using Tree method:



so, equation can be written as,

$T(n) = T(n-1) + a(1)$ → any constant

↓ one comparison per level

$\therefore \boxed{T(n) = a(n)}$

Since, work at each level,

→ 1+1+1+ --- (n-1) times

→ $a(n) + a(n-1) + a(1) + \dots a(n)$ (n-1) times

∵ $T.C = a(n-1) \Rightarrow a(n)$

Hence,

$\boxed{T.C = T(n) = a(n)}$ Ans

PAGE NO.:
DATE: / /

(2) Using Substitution Method:

Since, $T(n) = T(n-1) + O(1)$, $T(n) = O(1)$

$T(n) = T(n-1) + 1$

put $n = n-1$ then,

$T(n-1) = T((n-1)-1) + 1$

$T(n-1) = T(n-2) + 1$

put $n = n-2$ then,

$T(n-2) = T(n-3) + B \rightarrow$ const.

,

$\boxed{T(n) = O(n)}$

as linearly loop moving.

(3) Master's theorem:

∵ As per the eqn we will apply:

$T(n) = aT(n/b) + f(n)$

now by comparing by,

$T(n) = T(n-1) + O(1)$ we get,

$k=0$, $a=1$, $b=1$, $F(n) = O(1) =$ constant then,

by the given comparision we get,

$T(n) = O(n^{\log_b a} \cdot \log^k n) =$ ~~O(n)~~

$T(n) = O(n^{\log_1 1} \cdot \log^0 n)$

$\boxed{T(n) = O(n)}$ as $\boxed{\log_1 1 = 1}$

Hence proved.

Scanned with OKEN Scanner