Symbiosis Institute of Technology, Pune

Department of Computer Science and Engineering

Academic Year 2025-26

Design and Analysis of Algorithms– Lab

Batch 2023-27 - Sem V

Lab Assignment No:- 1	
Name of Student	Deepti Pal
PRN No.	23070122081
Batch	TY CSE (2023-27)
Class	A3
Academic Year & Semester	2025-26, Third Year, 5th Semester
Date of Submission	3 August 2025
Title of Assignment:	WAP to search an element using linear search, binary search (recursive) and analyze their complexity. **Problem Given:** • Create a student records file, with 5 columns: PRN, student_name, class, mobile_number, and marks. Let the PRN's be in ascending order. Write an interactive program to search student record given any of the data. Use linear search or binary search as applicable. The student record should have at least 25 records.
Theory: (Handwritten)	1. Compare Linear search and binary search algorithms.
Source code (Implementation Screenshot)	#include<stdio.h> #include<string.h> typedef struct { int prn; char name[50]; char class[10]; int mobile_no; int marks;

```
} Student;

//linear search
int linearSearch(Student arr[], int n, char* charTarget, int field) {
	for(int i=0; i<n; i++) {
		if(field == 2 && strcmp(arr[i].name, charTarget) == 0) {
			return i;
		}
		else if(field == 3 && strcmp(arr[i].class, charTarget)==0) {
			return i;
		}
	}
	return -1;
}

//binary search
int binarySearch(Student arr[], int n, int target, int field) {
	int low = 0;
	int high = n;
	while(low<=high) {
		int mid = (low + high)/2;
		int midVal = (field == 1)? arr[mid].prn :
			(field == 4)? arr[mid].mobile_no :
			arr[mid].marks;
		if(midVal == target) {
			return mid;
		}
		else if( target < midVal) {
			high = mid - 1;
		}
		else {
			low = mid + 1;
		}

	}
	return -1;
}

int main() {
	FILE *fp = fopen("TestFile_a1.txt", "r");
	char line[256];
	char *cell;
	char charTarget[50];
	int inputType, intTarget, result=-1;
	Student arr[30];

	if(fp == NULL) {
		printf("File Not found!");
		return 1;
	}

	fgets(line, sizeof(line), fp); // to skip the header line

	int c=0;
	while(fgets(line, sizeof(line), fp) && c < 30) {
```

```
            int s_no;
            sscanf(line, "%d %d %s %s %d %d",
                &s_no,
                &arr[c].prn,
                arr[c].name,
                arr[c].class,
                &arr[c].mobile_no,
                &arr[c].marks);
            c++;
        }


        printf("Enter your choice of input to search: \n");
        printf("1)Student PRN\n");
        printf("2)Student Name\n");
        printf("3)Student class\n");
        printf("4)Student Ph.no\n");
        printf("5)Student marks\n");
        scanf("%d", &inputType);

        switch(inputType) {
        case 1: {
            printf("Enter target prn: ");
            scanf("%d", &intTarget);
            result = binarySearch(arr, c, intTarget, 1);
            break;
        }
        case 2: {
            printf("Enter target name: ");
            scanf("%s", charTarget);
            result = linearSearch(arr, c, charTarget, 2);
            break;
        }
        case 3: {
            printf("Enter target class: ");
            scanf("%s", charTarget);
            result = linearSearch(arr, c, charTarget, 3);
            break;
        }
        case 4: {
            printf("Enter target Ph.no.: ");
            scanf("%d", &intTarget);
            result = binarySearch(arr, c, intTarget, 4);
            break;
        }
        case 5: {
            printf("Enter target marks: ");
            scanf("%d", &intTarget);
            result = binarySearch(arr, c, intTarget, 5);
            break;
        }
        }

        if(result<0) {
            printf("Details of target student not found !!");
```

```
        } else {
                printf("Details of the student are in row of s.no = %d",
result+1);
        }

        fclose(fp);
        return 0;
}
```

Input file:

main.c | TestFile_a1.txt

```
s.no    prn     student name    class   mobile no   marks
1       31      akshay          III     92681       59
2       32      krishna         IV      81723       52
3       33      ridhi           V       81919       88
4       34      ayush           III     62726       96
5       35      priya           VI      45258       48
6       36      radhika         III     81735       26
7       37      priyanka        IV      89121       58
8       38      dipesh          V       12340       52
9       39      aarushi         V       89119       37
10      40      nidhi           VI      99118       84
11      41      shubham         III     93214       75
12      42      mehak           IV      78632       64
13      43      vivek           V       81234       45
14      44      arjun           VI      74325       89
15      45      sneha           III     89674       67
16      46      ananya          IV      90123       79
17      47      dev         V       87456       33
18      48      tanvi           VI      82345       92
19      49      manav           III     85421       60
20      50      ishita          IV      87654       50
```

Output Screenshots

Run | Debug | Stop | { } Beautify

main.c | TestFile_a1.txt

```c
#include<stdio.h>
#include<string.h>


typedef struct  {
    int prn;
    char name[50];
    char class[10];
    int mobile_no;
    int marks;
} Student;

//linear search
int linearSearch(Student arr[], int n, char* charTarget, int
```

input

```
Enter your choice of input to search:
1)Student PRN
2)Student Name
3)Student class
4)Student Ph.no
5)Student marks
1
Enter target prn: 45
Details of the student are in row of s.no = 15

...Program finished with exit code 0
Press ENTER to exit console.
```

Run | Debug | Stop | { } Beautify | Langu

main.c | TestFile_a1.txt

```c
#include<stdio.h>
#include<string.h>


typedef struct  {
    int prn;
    char name[50];
    char class[10];
    int mobile_no;
    int marks;
} Student;

//linear search
int linearSearch(Student arr[], int n, char* charTarget, int field) {
```

input

```
Enter your choice of input to search:
1)Student PRN
2)Student Name
3)Student class
4)Student Ph.no
5)Student marks
2
Enter target name: shubham
Details of the student are in row of s.no = 11
```

to
arch
have

Run | Debug | Stop | { } Beautify

main.c | TestFile_a1.txt

```c
#include<stdio.h>
#include<string.h>


typedef struct  {
    int prn;
    char name[50];
    char class[10];
    int mobile_no;
    int marks;
} Student;

//linear search
int linearSearch(Student arr[], int n, char* charTarget
```

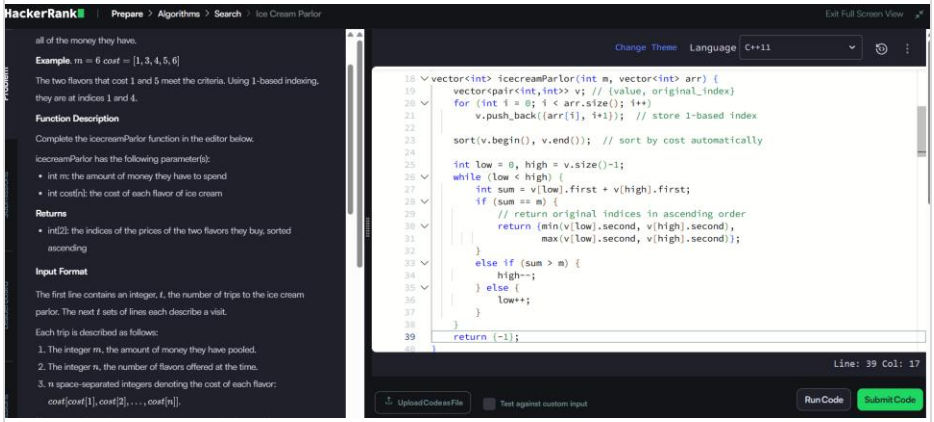
input

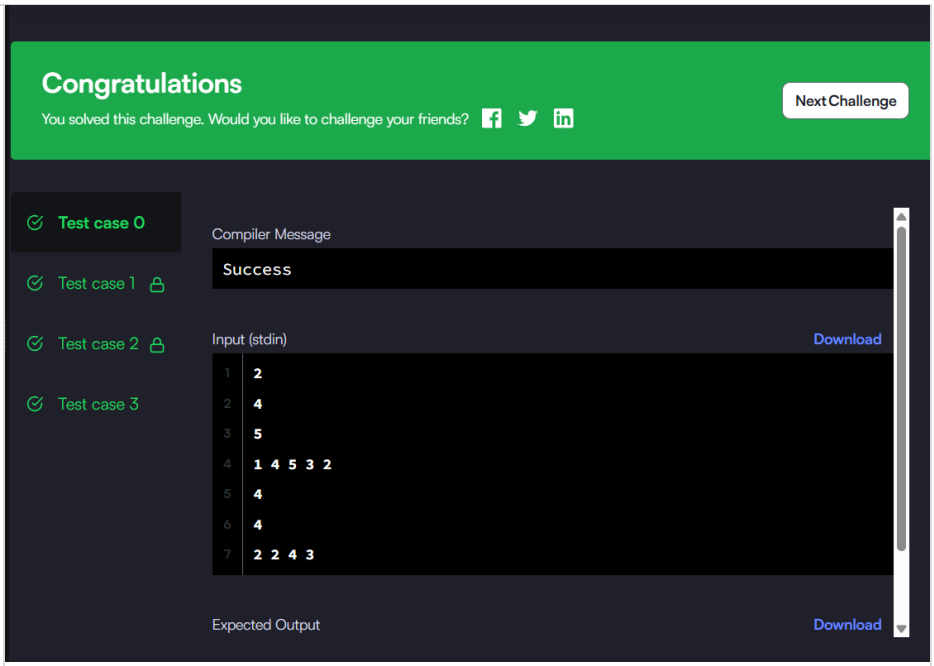
```
Enter your choice of input to search:
1)Student PRN
2)Student Name
3)Student class
4)Student Ph.no
5)Student marks
1
Enter target prn: 12
Details of target student not found !!

...Program finished with exit code 0
Press ENTER to exit console.
```

Problems Solved from Hacker Rank (Minimum 4)

1. https://www.hackerrank.com/challenges/icecream-parlor/problem?isFullScreen=true

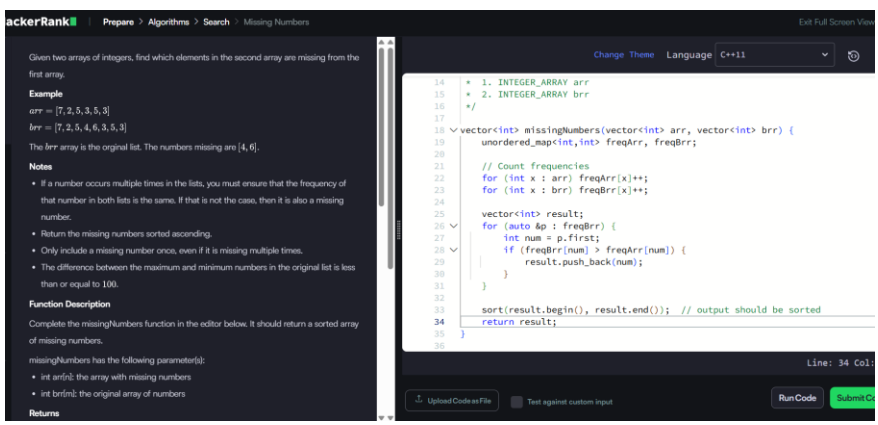
Solution Screenshot:

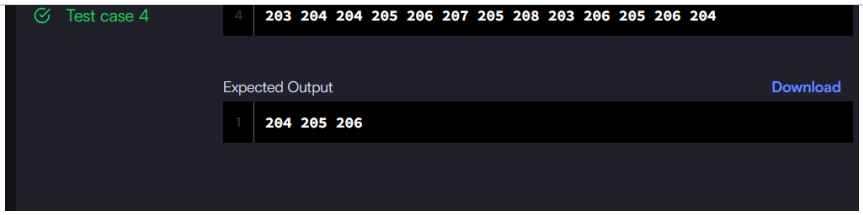




2. https://www.hackerrank.com/challenges/missing-numbers/problem?isFullScreen=true

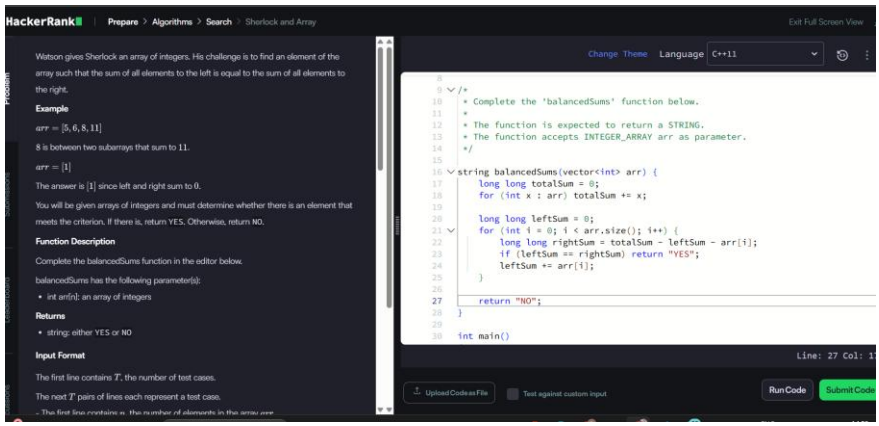
Solution Screenshot:

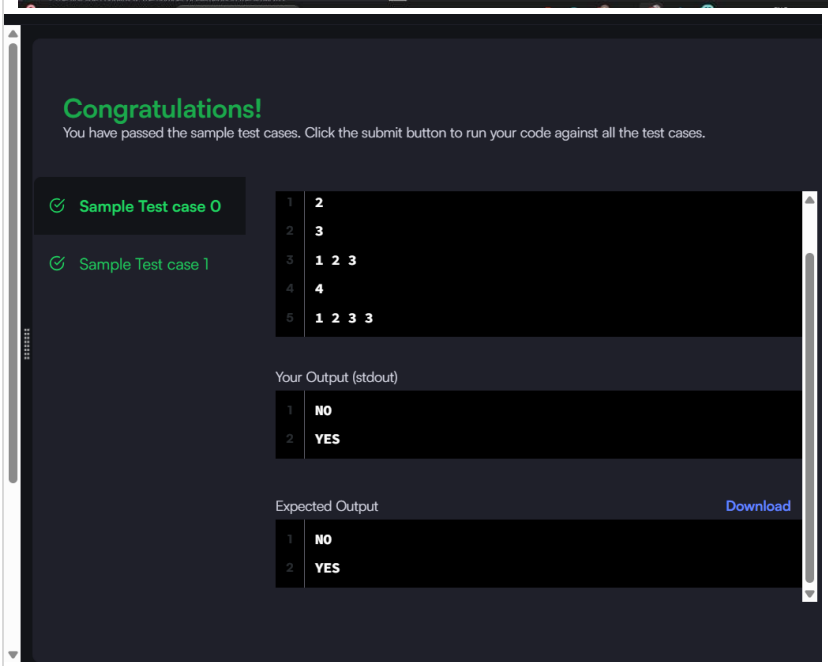




3. https://www. hackerrank.com/challenges/sherlock-and-array/problem?isFullScreen=true

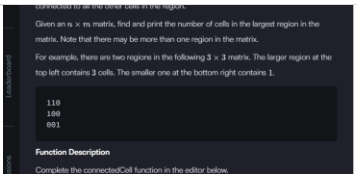
Solution Screenshot:





4. https://www.hackerrank.com/challenges/connected-cell-in-a-grid/problem?isFullScreen=true

Solution Screenshot:



Given an $n \times m$ matrix, find and print the number of cells in the largest region in the matrix. Note that there may be more than one region in the matrix.

For example, there are two regions in the following 3×3 matrix. The larger region at the top left contains 3 cells. The smaller one at the bottom right contains 1.

```
110
100
001
```

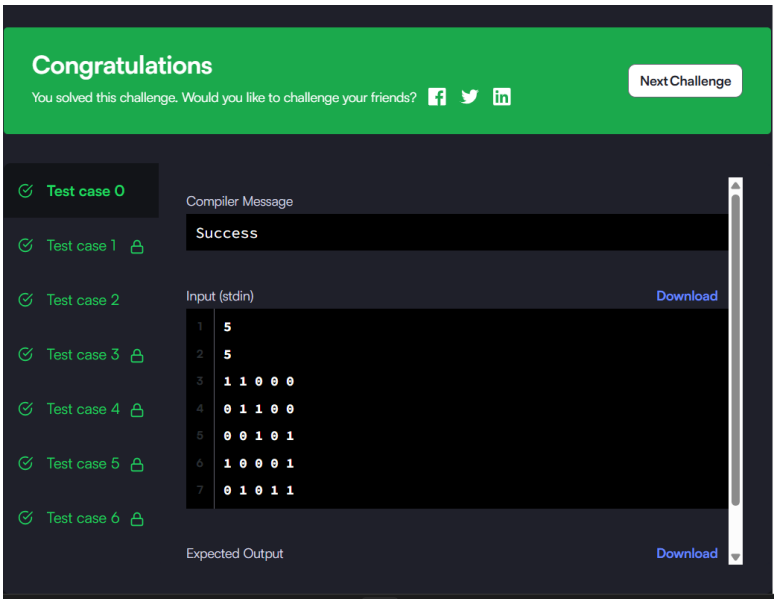
Function Description

Complete the connectedCell function in the editor below.

```
            int regionSize = dfs(matrix, i, j, visited);
            maxRegion = max(maxRegion, regionSize);
        }
    }
}
return maxRegion;
}

int main()
{
```

Line: 42 Col: 34

Upload Code as File | Test against custom input | Run Code | Submit Code



Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

- Test case 0
- Test case 1
- Test case 2
- Test case 3
- Test case 4
- Test case 5
- Test case 6

Compiler Message

```
Success
```

Input (stdin) — Download

```
5
5
1 1 0 0 0
0 1 1 0 0
0 0 1 0 1
1 0 0 0 1
0 1 0 1 1
```

Expected Output — Download