Student Portfolio

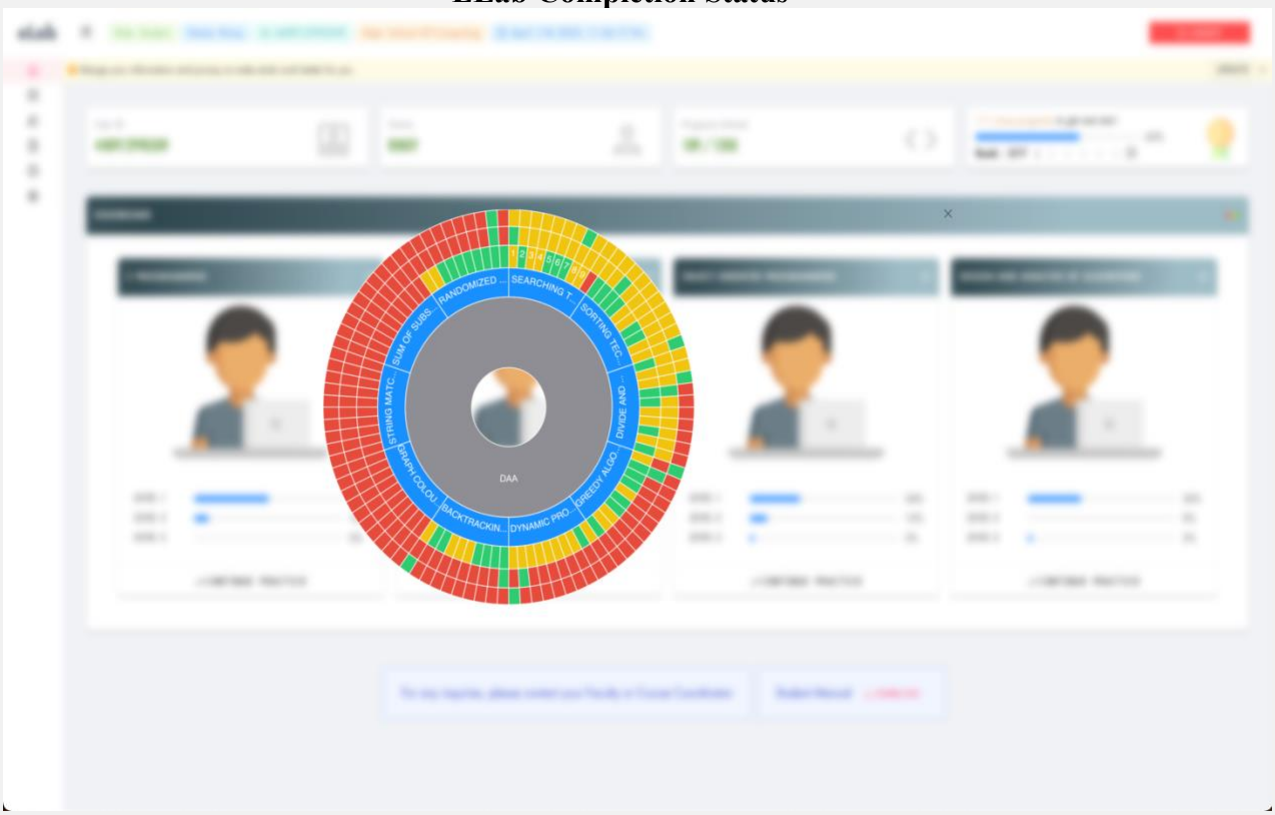
Rinoy Ramesh

Register Number: RA2311003012144
Mail ID: rr9921@srmist.edu.in
Department: CTECH
Year / Sem/ Section: 2nd/4th/O2

Subject Title: 21CSC206J Design and Analysis of Algorithms
Handled By: Dr.K.Geetha

Email: rr9921@srmist.edu.in
LinkedIn: linkedin.com/rinoyramesh
GitHub: github.com/rennnss
Portfolio Website (if any):

ELab Completion Status



Lab Experiment Completion status

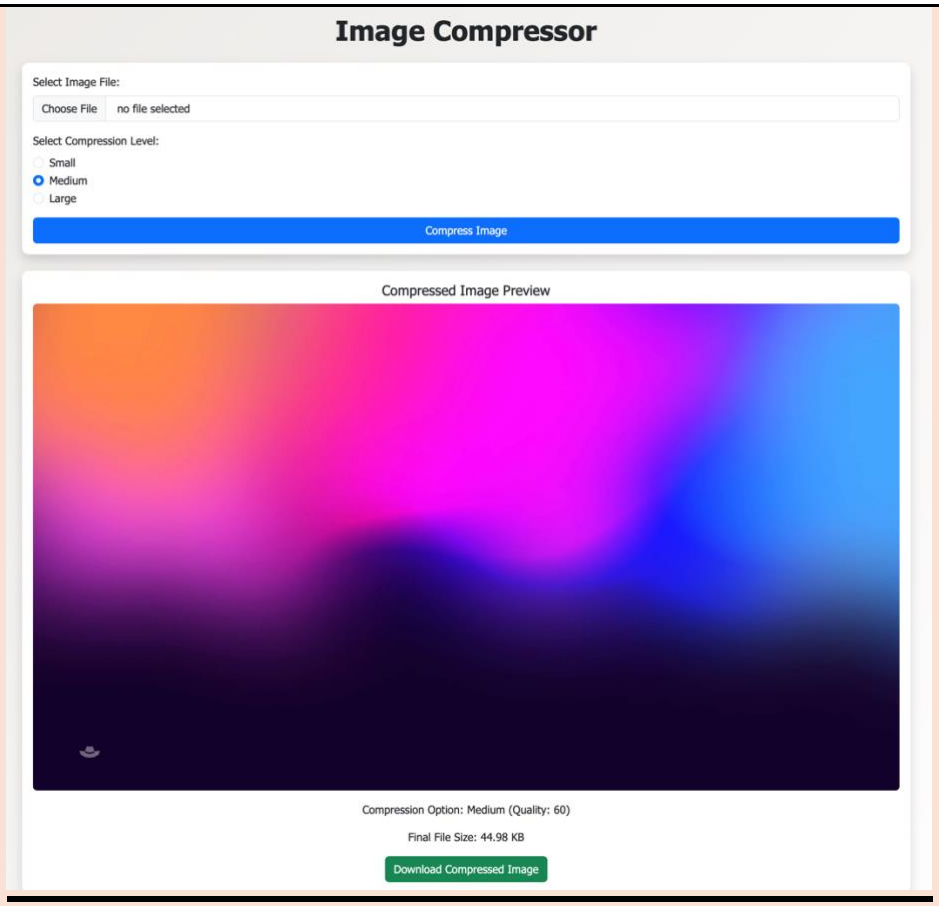
Name: Rinoy Ramesh

RA: RA2511003012144

EXP. No.	TITLE	Aim & Algorithm (1 Mark)	SUB TOTAL (10 Marks)					Time complexity analysis (3 Marks)	Dry run with sample I/P and O/P & Result (1 Mark)	VIVA (5 Marks)	TOTAL (20 Marks)	
			Basic Solution (2 Marks)	Modularity (2.5 Marks)	Readability (2.5 Marks)	Validation (2 Marks)	Scalability (1 Marks)					
1	a) Insertion Sort b) Bubble Sort	1	2	2	2	2	1	2	–	4	17	
2	Linear Search, Binary search	1	2	2	2	2	1	2	–	4	17	
3	Merge Sort	1	2	2	2	2	1	3		4	18	
4	Quick Sort	1	2	2	2	2	1	3		4	18	
5	Strassen Matrix Multiplication	1	2	2	2	2	1	3	1	4	19	
6	a) Finding Maximum and Minimum in an array b) Convex Hull Problem	1	2	2	2	2	1	1	1	3	15	
7	a) Huffman Coding b) Knapsack using Greedy	1	2	2	2	2	1	3	1	5	20	
8	Longest Common Subsequence	1	2	2	2	2	1	3	1	5	20.	
9	N Queen's Problem	1	2	2	2	2	1	3	1	4	19	
10	Travelling Salesman Problem	1	2	2	2	2	1	3	1	4	19	
11	Randomized Quick Sort	1	2	2	2	2	1	3	1	4	19	
12	String Matching Algorithms	1	2	2	2	2	1	3	1	5	19+1	

REAL TIME APPLICATION
(Image Compressor)

Name	Date Modified	Size
compressed-1.jpg	Today at 15:49	18 KB
Gradient3.png	Today at 01:34	5.9 MB



NPTEL/HOTS QUESTIONS SOLUTION

Chief's bot is playing an old DOS based game. There is a row of buildings of different heights arranged at each index along a number line. The bot starts at building 0 and at a height of 0. You must determine the minimum energy his bot needs at the start so that he can jump to the top of each building without his energy going below zero.

Units of height relate directly to units of energy. The bot's energy level is calculated as follows:

- If the bot's $botEnergy$ is less than the height of the building, his $newEnergy = botEnergy - (height - botEnergy)$
- If the bot's $botEnergy$ is greater than the height of the building, his $newEnergy = botEnergy + (botEnergy - height)$

Example

$arr = [2, 3, 4, 3, 2]$

Starting with $botEnergy = 4$, we get the following table:

```
botEnergy   height  delta
4               2      +2
6               3      +3
9               4      +5
14              3      +11
25              2      +23
48
```

That allows the bot to complete the course, but may not be the minimum starting value. The minimum starting $botEnergy$ in this case is 3.

Function Description

Complete the chiefHopper function in the editor below.

chiefHopper has the following parameter(s):

- int arr[n]: building heights

Returns

- int: the minimum starting $botEnergy$

Input Format

The first line contains an integer n, the number of buildings.

The next line contains n space-separated integers $arr[1]..arr[n]$, the heights of the buildings.

Constraints

- $1 \le n \le 10^5$
- $1 \le arr[i] \le 10^5$ where $1 \le i \le n$

Sample Input 0

```
5
3 4 3 2 4
```

Sample Output 0

```
4
```

Explanation 0

If initial energy is 4, after step 1 energy is 5, after step 2 it's 6, after step 3 it's 9 and after step 4 it's 16, finally at step 5 it's 28.
If initial energy were 3 or less, the bot could not complete the course.

Sample Input 1

```
3
4 4 4
```

Sample Output 1

```
4
```

Explanation 1

In the second test case if bot has energy 4, it's energy is changed by (4 - 4 = 0) at every step and remains 4.

Sample Input 2

```
3
1 6 4
```

Sample Output 2

```
3
```

Explanation 2

```
botEnergy  height  delta
3          1       +2
5          6       -1
4          4       0
4
```

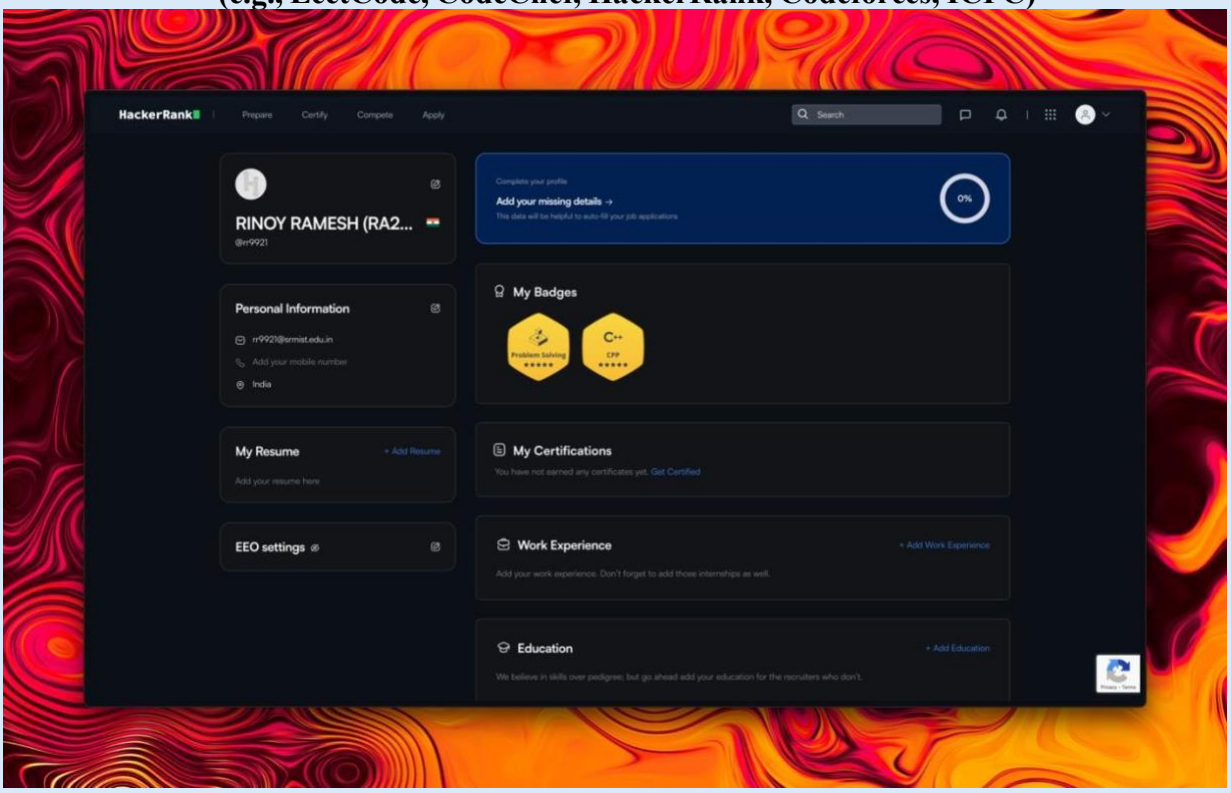
We can try lower values to assure that they won't work.

```
#include <iostream>
#include <vector>
using namespace std;

// Function to check if the given energy is sufficient
bool isEnough(int energy, const vector<int>& arr) {
long long currentEnergy = energy;
for (int height : arr) {
currentEnergy += currentEnergy - height;
if (currentEnergy < 0) return false;
}
return true;
}

// Binary search to find the minimum energy
int chiefHopper(vector<int>& arr) {
int low = 0, high = 1e6, result = high;
while (low <= high) {
int mid = (low + high) / 2;
if (isEnough(mid, arr)) {
result = mid;
high = mid - 1;  // Try to find a smaller valid energy
} else {
low = mid + 1;
}
}
return result;
}

int main() {
int n;
cin >> n;
vector<int> arr(n);
for (int& height : arr) {
cin >> height;
}

cout << chiefHopper(arr) << endl;
return 0;
}
```

CODING COMPETITIONS

Any notable rankings or achievements in coding contests

(e.g., LeetCode, CodeChef, HackerRank, Codeforces, ICPC)



Signature of the Student