Amanda has a string of lowercase letters that she wants to copy to a new string. She can perform the following operations with the given costs. She can perform them any number of times to construct a new string p:

- Append a character to the end of string p at a cost of 1 dollar.
- Choose any substring of p and append it to the end of p at no charge.

Given n strings $s[i]$, find and print the minimum cost of copying each $s[i]$ to $p[i]$ on a new line.

For example, given a string $s = abcabc$, it can be copied for 3 dollars. Start by copying a, b and c individually at a cost of 1 dollar per character. String $p = abc$ at this time. Copy $p[0:2]$ to the end of p at no cost to complete the copy.

Function Description

Complete the stringConstruction function in the editor below. It should return the minimum cost of copying a string.

stringConstruction has the following parameter(s):

- s: a string

Input Format

The first line contains a single integer n, the number of strings.
Each of the next n lines contains a single string, $s[i]$.

Constraints

- $1 \leq n \leq 5$
- $1 \leq |s[i]| \leq 10^5$

Subtasks

- $1 \leq |s[i]| \leq 10^3$ for 45% of the maximum score.

Output Format

For each string $s[i]$ print the minimum cost of constructing a new string $p[i]$ on a new line.

Sample Input

```
2
abcd
abab
```

Line: 12 Col: 16

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

You have earned 25.00 points!
You are now 67.2 points away from the 3rd star for your problem solving badge.
33% 132.8/200

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

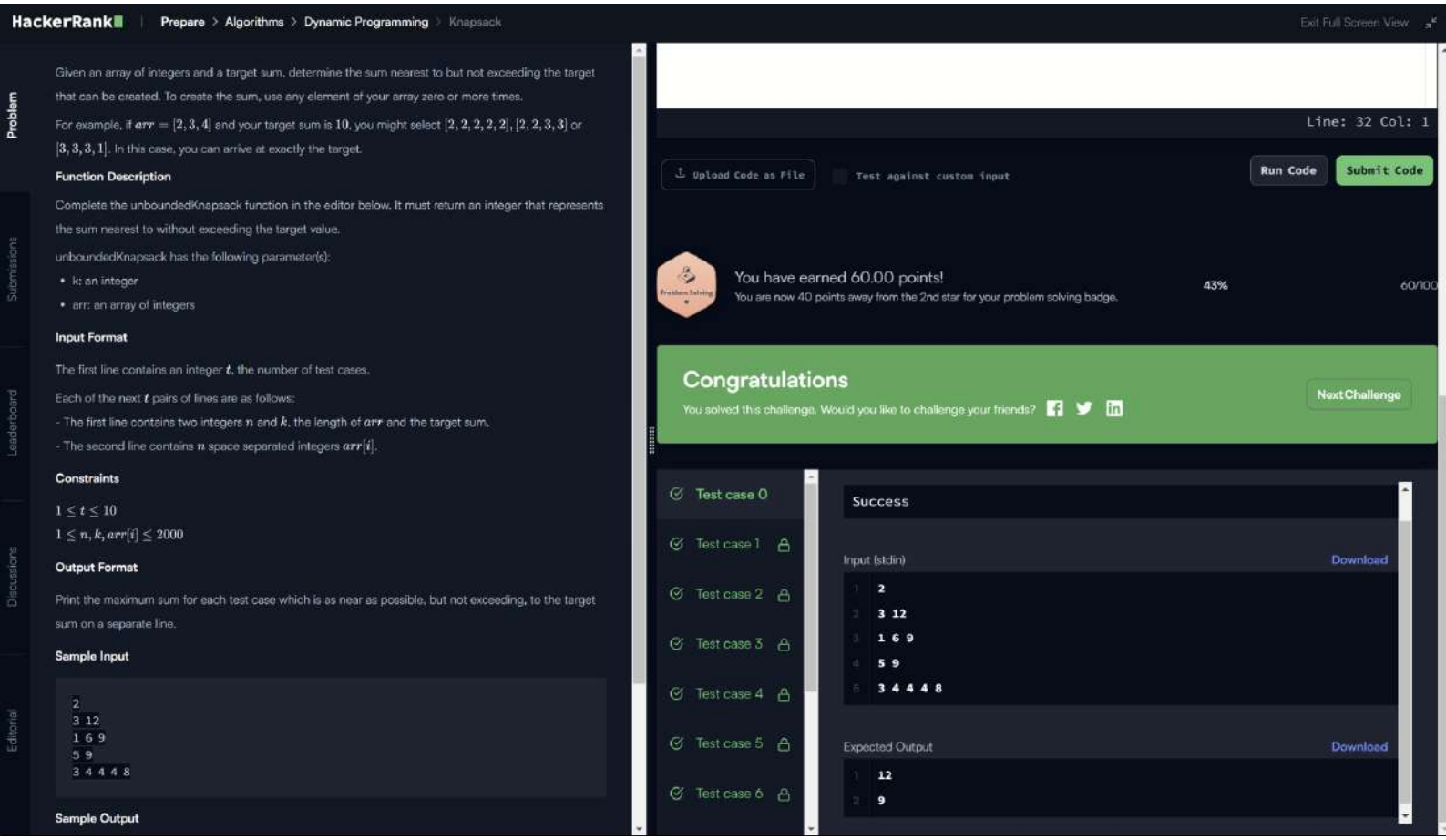
Compiler Message
Success

Input (stdin) Download

```
2
abcd
abab
```

Expected Output Download

```
4
2
```


HackerRank
Prepare > Algorithms > Dynamic Programming > Knapsack
Exit Full Screen View
Problem
Submissions
Leaderboard
Discussions
Editorial
Given an array of integers and a target sum, determine the sum nearest to but not exceeding the target that can be created. To create the sum, use any element of your array zero or more times.
For example, if $arr = [2, 3, 4]$ and your target sum is 10, you might select $[2, 2, 2, 2, 2]$, $[2, 2, 3, 3]$ or $[3, 3, 3, 1]$. In this case, you can arrive at exactly the target.
Function Description
Complete the unboundedKnapsack function in the editor below. It must return an integer that represents the sum nearest to without exceeding the target value.
unboundedKnapsack has the following parameter(s):
• k: an integer
• arr: an array of integers
Input Format
The first line contains an integer t, the number of test cases.
Each of the next t pairs of lines are as follows:
- The first line contains two integers n and k, the length of arr and the target sum.
- The second line contains n space separated integers $arr[i]$.
Constraints
$1 \leq t \leq 10$
$1 \leq n, k, arr[i] \leq 2000$
Output Format
Print the maximum sum for each test case which is as near as possible, but not exceeding, to the target sum on a separate line.
Sample Input
2
3 12
1 6 9
5 9
3 4 4 4 8
Sample Output
Line: 32 Col: 1
Upload Code as File
Test against custom input
Run Code
Submit Code
You have earned 60.00 points!
You are now 40 points away from the 2nd star for your problem solving badge.
43%
60/100
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
Success
Input (stdin)
Download
2
3 12
1 6 9
5 9
3 4 4 4 8
Expected Output
Download
12
9

HackerRank | Prepare > Algorithms > Greedy > Cloudy Day

Exit Full Screen View

Problem | Submissions | Leaderboard | Discussions | Editorial

Quibdó in Colombia is one among the cities that receive maximum rainfall in the world.

All year round, the city is covered in clouds. The city has many towns, located on a one-dimensional line. The positions and populations of each town on the number line are known to you. Every cloud covers all towns located at a certain distance from it. A town is said to be in darkness if there exists at least one cloud such that the town is within the cloud's range. Otherwise, it is said to be sunny.

The city council has determined that they have enough money to remove exactly one cloud using their latest technology. Thus they want to remove the cloud such that the fewest number of people are left in darkness after the cloud is removed. What is the maximum number of people that will be in a sunny town after removing exactly one cloud?

Note: If a town is not covered by any clouds, then it is already considered to be sunny, and the population of this town must also be included in the final answer.

Complete the function `maximumPeople` which takes four arrays representing the populations of each town, locations of the towns, locations of the clouds, and the extents of coverage of the clouds respectively, and returns the maximum number of people that will be in a sunny town after removing exactly one cloud.

Input Format

The first line of input contains a single integer n, the number of towns.

The next line contains n space-separated integers p_i. The i^{th} integer in this line denotes the population of the i^{th} town.

The next line contains n space-separated integers x_i denoting the location of the i^{th} town on the one-dimensional line.

```
if __name__ == "__main__":
    main()
53
```

Line: 53 Col: 1

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

Problem Solving

You have earned 45.00 points!

You are now 60 points away from the 3rd star for your problem solving badge.

40% 140/200

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

Success

Input (stdin) Download

```
2
10 100
5 100
1
4
1
```

Expected Output

```
110
```

Snipping Tool

Screenshot copied to clipboard and saved
Select here to mark up and share.

is the pivot element, p.

Constraints

- $1 \leq n \leq 1000$
- $-1000 \leq arr[i] \leq 1000$ where $0 \leq i < n$
- All elements are distinct.

Sample Input

```
STDIN       Function
-----       --------
5           arr[] size n =5
4 5 3 7 2   arr =[4, 5, 3, 7, 2]
```

Sample Output

```
3 2 4 5 7
```

Explanation

$arr = [4, 5, 3, 7, 2]$ Pivot: $p = arr[0] = 4$.
$left = \{\}; equal = \{4\}; right = \{\}$
$arr[1] = 5 > p$, so it is added to $right$.
$left = \{\}; equal = \{4\}; right = \{5\}$
$arr[2] = 3 < p$, so it is added to $left$.
$left = \{3\}; equal = \{4\}; right = \{5\}$
$arr[3] = 7 > p$, so it is added to $right$.
$left = \{3\}; equal = \{4\}; right = \{5, 7\}$
$arr[4] = 2 < p$, so it is added to $left$.
$left = \{3, 2\}; equal = \{4\}; right = \{5, 7\}$

Return the array $\{32457\}$.

The order of the elements to the left and right of **4** does not need to match this answer. It is only required that **3** and **2** are to the left of **4**, and **5** and **7** are to the right.

Test case 0
Test case 1
Test case 2
Test case 3
Test case 4

Compiler Message

```
Success
```

Input (stdin) Download

```
1 5
2 4 5 3 7 2
```

Expected Output Download

```
1 3 2 4 5 7
```