LAB-PRAC-07_LOOPS-ARR

Home Alone (p1v1d1)

Home Alone [10 marks]

Problem Statement
Your parents were not at home, so you decided to play with 2 box of cards. The first box had 5 cards, while the second box had 6 cards. Each card had an **integer** number on it. However, you made a mess, and mixed up cards from the two boxes. You know that your mother only remembered that the cards were sorted, and not which card belonged to which box. So, you decided to keep all the cards in 1 box, in sorted order. Can you do that before your parents come?

You will be given the numbers of the first box **as five integers sorted in non-decreasing order** in the first line of the input and the numbers of the second box **as six integers sorted in non-decreasing order** in the second line of the input. Numbers in each line will be separated by a single space.

Give your output as the list of integers from both boxes combined in non-decreasing order with a single space between 2 numbers. There should be no trailing spaces at the end of the output.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.
2. Be very careful, even though the evaluation may give you marks for extra spaces and newlines, the autograder will give you zero marks for any extra spaces or new lines.

HINTS:

1. You will require the use of arrays in this question.

EXAMPLE:
INPUT
```
1 3 5 6 8
2 4 6 7 9 10
```

OUTPUT:
```
1 2 3 4 5 6 6 7 8 9 10
```

Grading Scheme:
Total marks: **[10 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
1 3 5 6 8 2 4 6 7 9 10	1 2 3 4 5 6 6 7 8 9 10
1 3 5 7 9 2 4 6 8 10 12	1 2 3 4 5 6 7 8 9 10 12
1 3 3 3 3 2 3 3 3 3 3	1 2 3 3 3 3 3 3 3 3 3
-5 -3 -1 1 3 -4 -2 0 2 4 6	-5 -4 -3 -2 -1 0 1 2 3 4 6
1 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1 1
10 100 1000 10000 10001 2 4 6 7 8 9	2 4 6 7 8 9 10 100 1000 10000 10001

Arrangements with Arrays (p1v2d1)

Arrangements with Arrays [20 marks]

Problem Statement
On the first line of the input, you will be given a **strictly positive integer** n. We promise that n will always be less than 10. On the next two lines each, you will see a sequence of n **non-negative integers**. Store these two sequences in two arrays a and b.

Now, your output will have n lines. In the i-th line of the output, where i goes from 1 to n, you have to print the element a[i-1]. However, the number of times you print this integer will be different. If a[i-1] is odd, print it as many times as itâ€™s value else print it b[i-1] number of times. Be careful not to have extra spaces at the end of the lines or extra lines.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.
2. Be very careful, even though the evaluation may give you marks for extra spaces and newlines, the autograder will give you zero marks for any extra spaces or new lines.
3. 1, 3, 5 etc are considered an odd numbers, 0, 2, 4 etc are considered even numbers.

EXAMPLE:
INPUT
4
1 3 5 7
2 4 6 3

OUTPUT:
1

```
3 3 3
5 5 5 5 5
7 7 7 7 7 7 7
```

Grading Scheme:
Total marks: **[20 Points]**

There will be partial grading in this question. Printing each line correctly, in the correct order, carries some weightage. All lines have equal weightage i.e. if there are 4 lines in the expected output, each is worth 25% weightage. If there are 5 lines in the expected output, each is worth 20% weightage. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

All Test Cases (Visible + Hidden)

Input	Output

4
1 3 5 7
2 4 6 3
	1
	3 3 3
	5 5 5 5 5
	7 7 7 7 7 7 7

2
2 4
1 3
	2
	4 4 4

5
1 3 4 7 8
0 0 0 0 2
	1
	3 3 3
	
	7 7 7 7 7 7 7
	8 8

4
2 4 6 8
2 4 6 8
	2 2
	4 4 4 4
	6 6 6 6 6 6
	8 8 8 8 8 8 8 8

5
1 2 3 4 5
1 1 1 1 1
	1
	2
	3 3 3
	4
	5 5 5 5 5

5
5 4 3 2 1
0 0 0 0 0
	5 5 5 5 5
	
	3 3 3
	
	1

Overlapping Patterns (p1v3d1)

Overlapping Patterns [20 marks]

Problem Statement
Mr C likes figuring out patterns in sequences. He is very happy if he is able to find the pattern â€œ1 2 1â€ (without quotes) in the sequence. You will be given a sequence of integers in the input. The sequence contain only the numbers 0, 1 and 2. The sequence will be terminated by a -1 and this -1 is not a part of the sequence.

Mr C wants to figure out how many times does the pattern "1 2 1" (without quotes) appear in the pattern. Your output should simply be the number of time this pattern appears in the sequence you have been given.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.
2. The given sequence may have zero occurrences of the pattern. The sequence itself may be empty.
3. Note that, the occurrences of the pattern "1 2 1" (without quotes) can be overlapping as demonstrated in the following examples. However, in brief, we can appreciate this by looking at the sequence 1 2 1 2 1. This sequence actually contains two occurrences of the pattern 1 2 1. The third element of the sequence which is a 1, acts as the final element of one occurrence of the pattern 1 2 1 and also acts as the beginning of a new occurrence of the pattern 1 2 1. Thus, if we were to explicity bracket the occurrences, we would get (1 2 (1) 2 1)

HINT: You may require the use of flags in this question.

EXAMPLE 1:
INPUT
1 2 1 2 1 2 0 1 -1
OUTPUT:
2
Explanation: The first 3 numbers form 121. In addition, the 3rd to 5th numbers also form 121. The occurrences of the pattern are (1 2 (1) 2 1) 2 0 1

EXAMPLE 2:
INPUT
1 2 1 0 1 2 1 1 0 0 -1
OUTPUT:
2
Explanation: The first 3 numbers, as well as the 5th-7th numbers, form the pattern 121. Thus, the number of times the pattern appears is 2. The occurrences of the pattern are (1 2 1) 0 (1 2 1) 1 0 0

Grading Scheme:
Total marks: **[20 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 2 point and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
1 2 1 2 1 2 0 1 -1	2
1 2 1 0 1 2 1 1 0 0 -1	2
1 2 2 1 0 0 1 2 1 0 2 1 1 2 1 -1	2

1 2 1 2 1 1 1 2 1 2 1 -1	4
1 2 1 2 2 1 -1	1
1 2 1 2 1 2 1 2 2 1 -1	3

Lucky Draw (p2v1d1)

Lucky Draw [10 marks]

Problem Statement
You, along with your two best friends, go to a party, where a tile of 7 coupons is kept, one on top of the other. Each coupon has a unique non-negative integer number, which denotes how many gifts does someone win at the end of the party. You know that you and your friends will be the first to select the coupons, so you decide to cheat and keep the coupons with maximum, second maximum and third maximum at the top three places on the pile.

You will be given a list of 7 non-negative integers separated by a space as input. Write a code to find the position of the maximum element in the list, and swap it with the first element of the list. Then find the position of the second largest element of the list and swap it with the second element in the list. Finally, find the position of the third largest element in the list and swap it with the third element of the list. Output the resulting list with one space between two numbers. There should be no trailing spaces in your output.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.
2. Be very careful, even though the evaluation may give you marks for extra spaces and newlines, the autograder will give you zero marks for any extra spaces or new lines.

HINTS:

1. You will require the use of arrays in this question.

EXAMPLE:
INPUT
1 2 5 4 6 7 8

OUTPUT:
8 7 6 4 5 2 1
Explanation: The maximum element is 8, which gets swapped with the first element 1. The second maximum element is 7, which gets swapped with the second element 2. The third largest element is 6, which gets swapped with the third element 5.

Grading Scheme:
Total marks: **[10 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
1 2 5 4 6 7 8	8 7 6 4 5 2 1
1 4 6 2 3 5 8	8 6 5 2 3 4 1
70 60 50 4 3 2 1	70 60 50 4 3 2 1
700 800 60 40 30 2 10	800 700 60 40 30 2 10
10 20 30 80 70 60 50	80 70 60 10 20 30 50
4 5 10 11 13 12 14	14 13 12 11 5 10 4

Diamond Array (p2v2d1)

Diamond Array [20 marks]

Problem Statement
On the first line of the input, you will be given a **strictly positive odd integer** n. We promise that n will be less than 10. In the next line of the input, you will be given n integers, separated by a space. You have to print a beautiful design described below.

Caution

1. Be careful about extra/missing lines and extra/missing spaces. There should be no trailing spaces at the end of any line nor should there be any extra new lines.
2. Be very careful, even though the evaluation may give you marks for extra spaces and newlines, the autograder will give you zero marks for any extra spaces or new lines.

HINTS:

1. You may require the use of arrays in this question.

EXAMPLE:
INPUT
3
1 3 5

OUTPUT:
3
135
3

Explanation: There are 3 lines in the output since n = 3. The first line is a space followed by 3 (no trailing spaces after 3). The second line has 135 (no spaces between the numbers 1, 3, and 5 and no trailing spaces after 5). The last line has a space followed by 3 (no trailing spaces after 3).

Grading Scheme:
Total marks: **[20 Points]**

There will be partial grading in this question. Printing each line correctly, in the correct order, carries some weightage. All lines have equal weightage i.e. if there are 4 lines in the expected output, each is worth 25% weightage. If there are 5 lines in the expected output, each is worth 20% weightage. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
3 1 3 5	3 135 3
1 10	10
5 9 0 9 8 7	9 098 90987 098 9
7 1 0 1 0 1 0 1	0 101 01010 1010101 01010 101 0
5 8 8 8 8 8	8 888 88888 888 8
9 1 2 3 4 5 5 5 5 5	5 455 34555 2345555 123455555 2345555 34555 455 5

Linear Leap (p2v3d1)

Linear Leap [20 marks]

Problem Statement
You will be given as input, a list of **non-negative integers**. The list will end with a -1 but the -1 is not a part of the list. The value of a number in this list denotes its "jump size". For example, if the number at location i is 3, then the jump size at that element is 3. This means that all locations to the right of that location, within a distance of 3, are reachable from that location in a single jump. More precisely, in a single "jump" Mr C can go from location i to location i+1 or i+2 or i+3 if the number at location i is 3. The first number of the list is said to be at location 1 (not like arrays where the first element is subscript 0).

The number at location j will be said to be "reachable" from location i if it can be reached in any number of jump. Mr C starts by jumping on location 1 (this is his initial jump). He can now take any number of jumps, in any order, as allowed by the above rules. However, **all jumps are made to the right of the list i.e. no backward jumps**.

Print, as your output "YES" (without quotes) if the last location in the list (i.e. the location just before -1 is there) is reachable from location 1, else print "NO" (without quotes).

Caution

1. Be careful about extra/missing lines and extra/missing spaces in your output.
2. **Any hardcoding attempts will be given a straight 0**

EXAMPLE 1:
INPUT
3 1 2 0 0 -1

OUTPUT:
YES

Explanation: Since the number at location 1 is 3, the 2nd, 3rd and 4th locations can be reached in a single step from location 1. Now the last location (i.e. location number 5) can be reached in a single step from location 3 since the number stored at location 3 is 2. Thus, the last location before -1 is reachable from location 1.
Step 1: Land on location 1
Step 2: Jump to location 3
Step 3: Jump to location 5

Note that Mr C could have jumped to location 4 from location 1 as well but that would have been useless since he would have been stuck there since the number stored there is 0. However, the answer is YES since there does exist an alternate set of jumps that do take Mr C to the last location before the -1.

EXAMPLE 2:
INPUT
3 1 1 0 1 2 -1

OUTPUT:
NO

Explanation: Since the number at location 1 is 3, the locations 2, 3, 4 can be reached in a single jump from location 1. Now lets see what happens on jumping to each of these locations

1. The number at location 2 is 1 so only location 3 can be reached in a single jump from location 2

2. The number at location 3 is 2 so only location 4 can be reached in a single jump from location 3
3. The number at location 4 is zero so if we land there, we are stuck.

Thus, we see that at most, we can reach location 4 from location 1 in many ways (1 => 4, 1 => 3 => 4, 1 =>2 => 3 => 4) but we always get stuck there. Thus, there is no way we can reach the location 6 which is the last location before -1. So the answer is NO.

Grading Scheme:
Total marks: **[20 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
3 0 3 0 0 0 1 -1	NO
2 0 1 1 3 0 1 1 -1	YES
4 3 3 3 0 0 0 -1	YES
4 0 2 1 3 1 3 0 0 0 -1	YES
9 0 0 1 0 0 0 2 -1	YES
3 0 3 0 5 1 1 1 1 0 0 -1	NO

Candy Crush (p3v1d1)

Candy Crush [10 marks]

Problem Statement
Your friendly neighborhood shopkeeper has decided to reward your excellent performance at JEE and give you some candies. She gives you a list of 10 integers (not all may be positive) and tells you to select two locations in the list (i.e. two numbers between 1 and 10 -- the first number being smaller than or equal to the second number).

Once you tell her these two locations, the shopkeeper will to give as many candies as the sum of all the numbers occurring between the locations indicated by you (including the locations specified by you). Write an algorithm to calculate what is the maximum number of candies you can get this way. Your input will be a list of 10 integers and your output should be the maximum number of candies you can get, the starting location, and the ending location, all separated by a single space with no trailing spaces at the end.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.
2. Be careful that array locations start from zero whereas in your output, you should give locations that start from one, just as we do in day to day conversations.
3. We assure you that the numbers in the list will all lie between -1000 and 1000.

HINT: You will require the use of arrays in this question. Maintaining a running max variable may also help.

EXAMPLE:
INPUT
1 3 5 2 6 -7 1 1 -3 1

OUTPUT:
17 1 5
Explanation: If you select the locations 1 and 5, then you get will get 1+3+5+2+6 = 17 candies since the numbers between the first and fifth locations (including both ends) in the list are 1, 3, 5, 2, 6. You can verify that this is the maximum number of candies you can get.

Grading Scheme:
Total marks: **[10 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
1 3 5 2 6 -7 1 1 -3 1	17 1 5
-1 -1 -1 -1 1 -1 -1 -1 -1 -1	1 5 5
1 2 3 4 5 6 7 8 9 10	55 1 10
1 1 1 1 1 1 1 1 -11 22	22 10 10
1 2 3 -2 1 2 3 -2 1 2	11 1 10
1 -2 3 -2 30 -15 1 -20 1 1	31 3 5

Nested Safes (p3v2d1)

Nested Safes [20 marks]

Problem Statement
In the first week lab, we enclosed a number inside a safe. In the second week lab, we took numbers out of the safes and added them. This week we will build even stronger safes.

In the first line of the input you will be given a **strictly positive integer** n. We promise that n will be less than or equal to 10. In the next line we will give you n digits (as non-negative integers) separated by a space. You have to build a nested safe out of these digits as shown below.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.

2. There is a single space between any two digits on any line. However, there are no trailing spaces at the end of any line nor are there any trailing new lines.

HINTS: You will require the use of arrays for this question.

EXAMPLE:
INPUT

```
3
1 2 3
```

OUTPUT:

```
1 1 1 1 1 1
1 2 2 2 2 1
1 2 3 3 2 1
1 2 3 3 2 1
1 2 2 2 2 1
1 1 1 1 1 1
```

Explanation>: The safe has 3 layers since n = 3. The outer most layer is the first element of the list i.e. 1, the middle layer is the second element of the list i.e. 2 and the inner most layer is the third element of the list, i.e. 3. Note that there is a single space between any two digits but no trailing spaces at the end of any line, nor are there any trailing new lines after the last line.

Grading Scheme:
Total marks: **[20 Points]**

There will be partial grading in this question. Printing each line correctly, in the correct order, carries some weightage. All lines have equal weightage i.e. if there are 4 lines in the expected output, each is worth 25% weightage. If there are 5 lines in the expected output, each is worth 20% weightage. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

All Test Cases (Visible + Hidden)

Input	Output

3
1 2 3
	1 1 1 1 1 1
	1 2 2 2 2 1
	1 2 3 3 2 1
	1 2 3 3 2 1
	1 2 2 2 2 1
	1 1 1 1 1 1

1 1	1 1 1 1

4
1 0 0 1
	1 1 1 1 1 1 1 1
	1 0 0 0 0 0 0 1
	1 0 0 0 0 0 0 1
	1 0 0 1 1 0 0 1
	1 0 0 1 1 0 0 1

	1 0 0 0 0 0 0 1
	1 0 0 0 0 0 0 1
	1 1 1 1 1 1 1 1

2
0 0
	0 0 0 0
	0 0 0 0
	0 0 0 0
	0 0 0 0

6
1 2 3 3 2 1
	1 1 1 1 1 1 1 1 1 1 1 1
	1 2 2 2 2 2 2 2 2 2 2 1
	1 2 3 3 3 3 3 3 3 3 2 1
	1 2 3 3 3 3 3 3 3 3 2 1
	1 2 3 3 2 2 2 2 3 3 2 1
	1 2 3 3 2 1 1 2 3 3 2 1
	1 2 3 3 2 1 1 2 3 3 2 1
	1 2 3 3 2 2 2 2 3 3 2 1
	1 2 3 3 3 3 3 3 3 3 2 1
	1 2 3 3 3 3 3 3 3 3 2 1
	1 2 2 2 2 2 2 2 2 2 2 1
	1 1 1 1 1 1 1 1 1 1 1 1

5
1 0 0 0 1
	1 1 1 1 1 1 1 1 1 1
	1 0 0 0 0 0 0 0 0 1
	1 0 0 0 0 0 0 0 0 1
	1 0 0 0 0 0 0 0 0 1
	1 0 0 0 1 1 0 0 0 1
	1 0 0 0 1 1 0 0 0 1
	1 0 0 0 0 0 0 0 0 1
	1 0 0 0 0 0 0 0 0 1
	1 0 0 0 0 0 0 0 0 1
	1 1 1 1 1 1 1 1 1 1

Heros Arc (p3v3d1)

Hero's Arc [20 marks]

Problem Statement
Mr C is a big fan of movies. He is especially obsessed with the cinematic notion of the heroâ€™s arc, in which the heroâ€™s happiness keeps decreasing until the movie reaches a turning point. From this point onwards, the heroâ€™s happiness keeps increasing. In every movie that Mr C watches, he makes a note of the heroâ€™s happiness at every point of time. He then gives you his notes and asks you to determine if the movie contains a heroâ€™s arc or not.

You will be given a sequence of strictly positive numbers such that no two consecutive numbers will be equal (non-consecutive numbers may be equal). The sequence will end with a -1 which is not a part of the sequence. The numbers (excluding the last -1) indicate the hero's happiness at various successive points in the movie. Thus, happiness levels never remain the same, they either go up or down throughout the movie (to make the movie plot interesting).

You have to output "YES" (without the quotes) if the sequence (excluding the last -1) consists of a strictly decreasing sequence followed by a strictly increasing sequence such that the last element of the decreasing sequence is the same as the first element of the subsequent increasing sequence and "NO" otherwise (without the quotes).

In the second line of the output, in case your answer in the first line is YES, output the turning point of the movie, i.e. the last element of the decreasing sequence (which will also happen to be the first element of the subsequent increasing sequence). If your answer in the first line is NO, then print the last element of the sequence (i.e. the one just before the -1) in the second line.

Caution

1. Be careful about extra/missing lines and extra/missing spaces.
2. A strictly increasing sequence is considered a hero's sequence with the first element of the sequence being the turning point.
3. A strictly decreasing sequence is also considered a hero's sequence with the last element of the sequence (not including the -1) as the turning point.
4. A singleton sequence (a sequence with only one element before the -1) can be interpreted as an increasing or a decreasing sequence. It will not change the correct answers to this question.

HINT: You may require the use of flags in this question.

EXAMPLE 1:
INPUT
7 6 5 4 10 11 12 -1

OUTPUT:
YES
4
Explanation: The strictly decreasing sequence is 7 6 5 4 and the strictly increasing sequence is 4 10 11 12. The element 4 is in common to them.

EXAMPLE 2:
INPUT
4 8 1 3 -1

OUTPUT:
NO
3
Explanation: The sequence first increases 4 8 then decreases 8 1 then increases again 1 3. 3 is output in the second line since it is the last element of the list (excluding the final -1).

Grading Scheme:
Total marks: **[20 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
7 6 5 4 10 11 12 -1	YES 4
9 6 4 7 8 10 5 -1	NO 5

9 7 5 3 1 2 4 6 8 11 -1	YES 1
9 7 5 3 1 2 6 8 11 13 10 12 -1	NO 12
2 3 4 8 10 -1	YES 2
9 1 2 4 6 7 10 12 -1	YES 1

Linear Loopy Maze (p4v1d1)

Linear Loopy Maze [10 marks]

Problem Statement
Mr C is caught in a linear maze. Help him find out if he can get out of the maze or not. In the first line of the input you will be given a **strictly positive integer** n. We promise that n will be less than or equal to 20. In the next line, we will give you n **non-negative** integers, separated by a single space. Store these numbers in an array. Lets call this array maze.

Mr C will land on the first element of maze i.e. subscript 0 of the array. The value maze[0] at that subscript will tell Mr C which subscript to land next i.e. in the next step Mr C will land on the maze[0]-th subscript of the array maze. The value stored there will tell him which subscript to land next. The exit of this maze is the last location of the array i.e. the subscript (n-1).

1. If following the above procedure, Mr C ever lands on the exit subscript, print "EXIT" (without quotes) followed by a space followed by the number of steps it took to reach the exit location for the first time.
2. If following the above procedure Mr C ever lands on a subscript which tells him to go to an illegal subscript in the array, print "BAD" (without quotes) followed by a space followed by the bad subscript.
3. If following the above procedure Mr C is never asked to go to an illegal subscript, but he can never hope to land on the exit subscript either, print "TRAP" (without quotes).

Caution

1. Be careful about spelling errors, extra/missing lines and extra/missing spaces.
2. When calculating the number of steps taken to reach the exit subscript (if reachable at all), the act of landing on the subscript 0 of the maze at the very beginning is to be considered the first step.
3. If Mr C ever reaches the exit subscript, he simply exits, he does not have to read the value at the exit subscript and continue this process anymore.

EXAMPLE 1:
INPUT
4
1 0 3 2

OUTPUT:

TRAP

Explanation: Mr C will land on subscript 0 in the array which contains 1 and will tell him to go to subscript 1. But there he will be told to go to subscript 0 and so he will get into an infinite loop and never hope to reach the exit subscript 3.

EXAMPLE 2:
INPUT
6
1 2 3 4 5 0

OUTPUT:
EXIT 6

Explanation:
Step 1: land on subscript 0
Step 2: land on subscript 1
Step 3: land on subscript 2
Step 4: land on subscript 3
Step 5: land on subscript 4
Step 6: land on subscript 5
and subscript 5 is the exit subscript

EXAMPLE 2:
INPUT
5
1 22 4 2 0

OUTPUT:
BAD 22

Explanation: In the third step Mr C will be asked to go to the subscript 22 which is out of bounds of the array.

Grading Scheme:
Total marks: **[10 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
9 2 3 5 0 54 1 3 5 3	TRAP
10 6 7 8 1 0 3 3 2 9 5	EXIT 8
12 4 1 2 3 7 1 2 45 3 1 2 3	BAD 45
7 5 12 11 3 11 3 11	TRAP

5 4 2 3 1 4	EXIT 2
1 2	EXIT 1

Histogram Heights (p4v2d1)

Histogram Heights [20 marks]

Problem Statement
In the first line of the input, you will be given a **strictly positive integer** n that is less than or equal to 20. In the next line you will be give n **non-negative integers** all separated by a space. You have to draw a histogram using the character O (capital O) and spaces as shown below. The histogram will have n bars and their heights will be dictated by the n integers given to you.

Caution

1. There is no extra space at the end of every line, nor are there extra lines.
2. Be very careful, even though the evaluation may give you marks for extra spaces and newlines, the autograder will give you zero marks for any extra spaces or new lines.

HINT: You will require the use of arrays in this question.

EXAMPLE:
INPUT
4
1 3 5 2

OUTPUT:
```
O
Â Â  O
Â  O O
Â  O O O
O O O O
```
Explanation: There are 4 vertical bars since there are 4 elements in the list. There is a single space between two of the bars. There are no trailing spaces at the end of any of the lines nor are there extra lines at the end of the input.

Grading Scheme:
Total marks: **[20 Points]**

There will be partial grading in this question. Printing each line correctly, in the correct order, carries some weightage. All lines have equal weightage i.e. if there are 4 lines in the expected output, each is worth 25% weightage. If there are 5 lines in the expected output, each is worth 20% weightage. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

All Test Cases (Visible + Hidden)

Input	Output
4 1 3 5 2	O O O O O O O O O O O
7 3 2 1 0 1 2 3	O O O O O O O O O O O O
5 5 1 2 4 3	O O O O O O O O O O O O O O O
10 3 3 3 3 3 3 3 3 3 3	O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O
7 0 0 0 0 4 0 0	O O O O
18 0 1 2 0 1 2 3 0 1 2 3 4 0 1 2 3 4 5	O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O O

Changing Times (p4v3d1)

Changing times [20 marks]

Problem Statement
Mr. C likes change, but steady change. You will be given a list of **non-negative** integers as input. The list will use -1 as the delimiter i.e. the -1 will not be a part of the list. Your job is to find the length of the longest sequence of consecutive integers in the list that forms an AP. We promise that the list will contain at least one element.

Caution

1. Be careful about extra/missing lines and extra/missing spaces in your output.
2. A sequence a0, a1, a2, ..., ak is said to be in an AP if, for some integer p, we have aj - a(j-1) = p , for all j = 1, 2, ..k
3. A sequence of one or two integers is always in AP by default.

HINT: You may benefit from the use of flags in this question.

EXAMPLE:
INPUT
1 4 7 11 17 13 9 5 -1

OUTPUT:
4
Explanation: The largest sequence in AP in this list is 17 13 9 5

Grading Scheme:
Total marks: **[20 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

All Test Cases (Visible + Hidden)

Input	Output
1 4 7 11 17 13 9 5 -1	4
4 -1	1
9 7 5 3 1 2 4 6 8 10 13 16 19 22 25 27 -1	6
9 7 5 3 1 2 4 6 8 11 14 -1	5
13 8 3 5 7 9 11 29 27 25 23 -1	5
13 14 15 16 17 14 11 8 5 2 -1	6