

X


<https://swayam.gov.in>

https://swayam.gov.in/nc_details/NPTEL

reviewer4@nptel.iitm.ac.in ▾

[NPTEL \(https://swayam.gov.in/explorer?ncCode=NPTEL\)](https://swayam.gov.in/explorer?ncCode=NPTEL) » **Programming, Data Structures And Algorithms**
Using Python (course)

Announcements (announcements)

About the Course (https://swayam.gov.in/nd1_noc19_cs40/preview) Ask a Question (forum)

Progress (student/home) Mentor (student/mentor)

Week 3 Programming Assignment

Due on 2019-08-22, 23:59 IST

Write three Python functions as specified below. Paste the text for all three functions together into the submission window. Your function will be called automatically with various inputs and should return values as specified. Do not write commands to read any input or print any output.

- You may define additional auxiliary functions as needed.
- In all cases you may assume that the value passed to the function is of the expected type, so your function does not have to check for malformed inputs.
- For each function, there are normally some public test cases and some (hidden) private test cases.
- "Compile and run" will evaluate your submission against the public test cases.
- "Submit" will evaluate your submission against the hidden private test cases. There are 10 private test cases, with equal weightage. You will get feedback about which private test cases pass or fail, though you cannot see the actual test cases.
- Ignore warnings about "Presentation errors".

1. Write a function `expanding(l)` that takes as input a list of integer `l` and returns `True` if the absolute difference between each adjacent pair of elements strictly increases.

Here are some examples of how your function should work.

```
>>> expanding([1, 3, 7, 2, 9])
True
```

 Course
outline

 How to access
the portal

 Week 1:
Introduction

Week 1 Quiz

 Week 2: Basics
of Python

Week 2 Quiz

 Week 2
Programming
Assignment

 Week 3: Lists,
inductive
function
definitions,
sorting

 Week 3
Programming
Assignment

○ Week 3
Programming
Assignment
([noc19_cs40/progassignment?
name=91](https://onlinecourses.nptel.ac.in/noc19_cs40/progassignment?name=91))

Week 4: Sorting,
Tuples,
Dictionaries,
Passing
Functions, List
Comprehension

Week 4 Quiz

Week 4
Programming
Assignment

Week 5:
Exception
handling,
input/output, file
handling, string
processing

Week 5
Programming
Assignment

Week 6:
Backtracking,
scope, data
structures;
stacks, queues
and heaps

Week 6 Quiz

Week 7:
Classes, objects
and user
defined
datatypes

Week 7 Quiz

Week 8:
Dynamic
programming,
wrap-up

Week 8
Programming
Assignment

Explanation: Differences between adjacent elements are $3 - 1 = 2$, $7 - 3 = 4$, $7 - 2 = 5$, $9 - 2 = 7$.

```
>>> expanding([1, 3, 7, 2, -3])
False
```

Explanation: Differences between adjacent elements are $3 - 1 = 2$, $7 - 3 = 4$, $7 - 2 = 5$, $2 - (-3) = 5$, so not strictly increasing.

```
>>> expanding([1, 3, 7, 10])
False
```

Explanation: Differences between adjacent elements are $3 - 1 = 2$, $7 - 3 = 4$, $10 - 7 = 3$, so not (strictly) increasing.

- Write a function `accordian(l)` that takes as input a list of integer `l` and returns `True` if the absolute difference between each adjacent pair of elements alternates between increasing strictly and decreasing strictly.

Here are some examples of how your function should work.

```
>>> accordian([1, 5, 1])
False
```

Explanation: Differences between adjacent elements are $5 - 1 = 4$, $5 - 1 = 4$, which are equal.

```
>>> accordian([1, 5, 2, 8, 3])
True
```

Explanation: Differences between adjacent elements are $5 - 1 = 4$, $5 - 2 = 3$, $8 - 2 = 6$, $8 - 3 = 5$, so the differences decrease, increase and then decrease.

```
>>> accordian([-2, 1, 5, 2, 8, 3])
True
```

Explanation: Differences between adjacent elements are $1 - (-2) = 3$, $5 - 1 = 4$, $5 - 2 = 3$, $8 - 2 = 6$, $8 - 3 = 5$, so the differences increase, decrease, increase and then decrease.

```
>>> accordian([1, 5, 2, 8, 1])
False
```

Explanation: Differences between adjacent elements are $1 - (-2) = 3$, $5 - 1 = 4$, $5 - 2 = 3$, $8 - 2 = 6$, $8 - 1 = 7$, so the differences increase, decrease, increase and then increase again.

- A square $n \times n$ matrix of integers can be written in Python as a list with `n` elements, where each element is in turn a list of `n` integers, representing a row of the matrix. For instance, the matrix

**Download
videos**

Text Transcripts

**Online
Programming
Test - Sample**

**Online
Programming
Test 1, 26 Sep
2019, 09:30-
11:30**

**Online
Programming
Test 2, 26 Sep
2019, 20:00-
22:00**

row of the matrix. For instance, the matrix

```
1 2 3
4 5 6
7 8 9
```

would be represented as `[[1,2,3], [4,5,6], [7,8,9]]`.

Write a function `rotate(m)` that takes a list representation `m` of a square matrix as input, and returns the matrix obtained by rotating the original matrix clockwise by 90 degrees. For instance, if we rotate the matrix above, we get

```
7 4 1
8 5 2
9 6 3
```

Your function should *not* modify the argument `m` provided to the function `rotate()`.

Here are some examples of how your function should work.

```
>>> rotate([[1,2],[3,4]])
[[3, 1], [4, 2]]
```

Explanation:

```
1 2   becomes   3 1
3 4           4 2
```

```
>>> rotate([[1,2,3],[4,5,6],[7,8,9]])
[[7, 4, 1], [8, 5, 2], [9, 6, 3]]
```

Explanation:

```
1 2 3   becomes   7 4 1
4 5 6           8 5 2
7 8 9           9 6 3
```

```
>>> rotate([[1,1,1],[2,2,2],[3,3,3]])
[[3, 2, 1], [3, 2, 1], [3, 2, 1]]
```

Explanation:

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

Sample Test Cases

	Input	Output
Test Case 1	<code>expanding([11, 35, 77, 21, 98])</code>	True
Test Case 2	<code>expanding([11, 38, 79, 25, -36])</code>	True
Test Case 3	<code>expanding([11, 33, 77, 100])</code>	False
Test Case 4	<code>expanding([-1, 2, -3, 4, -5, 6, -7, 8, -9, 10, -11, 12])</code>	True
Test Case 5	<code>expanding([-1, 2, -3, 4, -5, 6, -7, 8, -9, 10, -11, -32])</code>	False
Test Case 6	<code>accordian([23, 44, 22, 1, 26, 10])</code>	True
Test Case 7	<code>accordian([23, 44, 22, 1, 5, 1])</code>	False
Test Case 8	<code>accordian([1, 10, 2, 11, 3, 12, 4, 13, 5, 14, 6])</code>	True
Test Case 9	<code>accordian([1, 10, 2, 11, 3, 12, 4, 13, 5, 14, 23])</code>	False
Test Case 10	<code>accordian([12, 55, 22, 8, 40])</code>	True
Test Case 11	<code>rotate([[1, 1, 1, 1], [2, 2, 2, 2], [3, 3, 3, 3], [4, 4, 4, 4]])</code>	<code>[[4, 3, 2, 1], [4, 3, 2, 1], [4, 3, 2, 1], [4, 3, 2, 1]]</code>
Test Case 12	<code>rotate([[1, 1, 1, 1, 1], [2, 2, 2, 2, 2], [3, 3, 3, 3, 3], [4, 4, 4, 4, 4], [5, 5, 5, 5, 5]])</code>	<code>[[5, 4, 3, 2, 1], [5, 4, 3, 2, 1], [5, 4, 3, 2, 1], [5, 4, 3, 2, 1], [5, 4, 3, 2, 1]]</code>
Test Case 13	<code>rotate([[1, 1, 1, 1, 1, 1], [2, 2, 2, 2, 2, 2], [3, 3, 3, 3, 3, 3], [4, 4, 4, 4, 4, 4], [5, 5, 5, 5, 5, 5], [6, 6, 6, 6, 6, 6]])</code>	<code>[[6, 5, 4, 3, 2, 1], [6, 5, 4, 3, 2, 1], [6, 5, 4, 3, 2, 1], [6, 5, 4, 3, 2, 1], [6, 5, 4, 3, 2, 1], [6, 5, 4, 3, 2, 1]]</code>

Test Case 14	<code>rotate([[1,1,1,1,1,1,1],[2,2,2,2,2,2,2],[3,3,3,3,3,3,3],[4,4,4,4,4,4,4],[5,5,5,5,5,5,5],[6,6,6,6,6,6,6],[7,7,7,7,7,7,7]])</code>	<code>[[7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1], [7, 6, 5, 4, 3, 2, 1]]</code>
Test Case 15	<code>rotate([[1,1,1,1,1,1,1,1],[2,2,2,2,2,2,2,2],[3,3,3,3,3,3,3,3],[4,4,4,4,4,4,4,4],[5,5,5,5,5,5,5,5],[6,6,6,6,6,6,6,6],[7,7,7,7,7,7,7,7],[8,8,8,8,8,8,8,8]])</code>	<code>[[8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1], [8, 7, 6, 5, 4, 3, 2, 1]]</code>
Test Case 16	<code>expanding([1,3,7,2,9])</code>	True
Test Case 17	<code>expanding([1,3,7,2,-3])</code>	False
Test Case 18	<code>expanding([1,3,7,10])</code>	False
Test Case 19	<code>accordian([1,5,1])</code>	False
Test Case 20	<code>accordian([1,5,2,8,3])</code>	True
Test Case 21	<code>accordian([-2,1,5,2,8,3])</code>	True
Test Case 22	<code>accordian([1,5,2,8,1])</code>	False
Test Case 23	<code>rotate([[1,2],[3,4]])</code>	<code>[[3, 1], [4, 2]]</code>
Test Case 24	<code>rotate([[1,2,3],[4,5,6],[7,8,9]])</code>	<code>[[7, 4, 1], [8, 5, 2], [9, 6, 3]]</code>
Test Case 25	<code>rotate([[1,1,1],[2,2,2],[3,3,3]])</code>	<code>[[3, 2, 1], [3, 2, 1], [3, 2, 1]]</code>

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.

