Review Test Submission: Self-check: L6-Numerical Analysis with NumPy

User	Rakshit Yadav
Course	BMAN73701 Programming in Python for Business Analytics 2023-24 1st Semester
Test	Self-check: L6-Numerical Analysis with NumPy
Started	29/11/23 09:10
Submitted	29/11/23 09:18
Status	Completed
Attempt Score	50 out of 80 points
Time Elapsed	7 minutes
Results Displayed	All Answers, Submitted Answers, Correct Answers, Feedback

Question 1

10 out of 10 points

```
import [module] as np

A = np.[mod2].rand(2,5)

# Compute the mean of each column

np.[fun1](A, axis = [ax1])

# Compute the minimum of each row

np.[fun2](A, axis = [ax2])
```

Specified Answer for: module	✓ numpy
Specified Answer for: mod2	✓ random
Specified Answer for: fun1	✓ mean
Specified Answer for: ax1	✓ 0
Specified Answer for: fun2	✓ min

← OK

Specified Answer for: ax2 ✓ 1

Correct Answers for: module		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	numpy	
Correct Answers for: mod2		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	random	
Correct Answers for: fun1		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	mean	
Correct Answers for: ax1		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	0	
Correct Answers for: fun2		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	min	
Correct Answers for: ax2		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	1	

Response Feedback: Perfect!

Question 2

0 out of 10 points

```
# What is returned by the following code ?
X = np.array([[1,2,3],[4,5,6]])
X[np.logical_and(X >= 3, X <= 5)]
```

Selected Answer: ✗ An error: You cannot index a matrix with the result of np.logical_and

Answers: ✓ A vector containing the elements of A that are larger or equal to 3 and smaller and equal to 5

An error: You cannot index a matrix with the result of np.logical_and

A matrix containing the elements of A that are larger or equal to 3 and smaller and equal to 5

Response Feedback: Incorrect ! Remember about boolean indexing and what happens when you index a matrix with a boolean matrix of the same shape. Revise the lecture slides.

Question 3 10 out of 10 points

```
# If A is a NumPy matrix, what is computed by the following ?

A * A
```

Selected Answer: A matrix like A but each element is squared (like A**2)

Answers:

An error: we cannot multiply matrices with other matrices in this way.

A matrix like A but each element is squared (like A**2)

The matrix product of A with itself.

A new matrix that is made of A copies of A

Response Feedback: Correct ! All mathematical operators operate element-wise. Matrix multiplication is done with the function np.`dot(A,A)`

Question 4 0 out of 10 points

We have a vector x of length 5 and a matrix A of shape (5,5). We want to sum each element of x_i to the corresponding element A_{ij} of

each column *j* of *A*. How can we do that in Python?

Selected Answers: `x + A`

Answers:
- `np.sum(x.reshape((5,1)), A)`
- `np.sum(x, A)`
- `x + A`
- `x.reshape((5,1)) + A`

Response Feedback: Incorrect! `np.sum()` will just sum up everything.

Also revise the broadcasting rules on how to match a vector with the columns of a matrix.

Finally, even if Python does not give an error, the code may not be doing what you want. In this case, *x* + *A* would sum x to each row of A, however, we wish to sum x to each column of A.

Question 5

0 out of 10 points

What are the benefits of using NumPy instead of lists and loops?

Selected Answers:
- Numpy vector/matrix operations are often shorter to write (less code).
- Numpy vector/matrix operations are faster than loops.

Answers:
- Numpy vector/matrix operations are easier to read.
- Numpy vector/matrix operations are more professional.
- Numpy vector/matrix operations are often shorter to write (less code).
- You should never use lists.
- Numpy already implements many mathematical operations.
- Numpy vector/matrix operations are easier to read.
- Numpy vector/matrix operations are faster than loops.
- You should never use for-loops

Response Feedback: Incorrect!

There are cases where for-loops and lists are more appropriate and easier to read than NumPy operations. However, for mathematical calculations, NumPy operations are often faster

and shorter to write and NumPy already implements many of them, so that you do not have to write them yourself.

Question 6

10 out of 10 points

If `a` is a numpy array then `a * 2` computes

Selected Answer: A new array where each element of **a** is multiplied by 2.

Answers:
- Exactly the same as **a.append(a)**
- A new array that is the result of appending **a** to itself.
- A new array where each element of **a** is multiplied by 2.
- A new array where each element of **a** is squared.

Response Feedback: You are right. The operator ***** applied to numpy arrays performs element-wise multiplication.

Question 7

10 out of 10 points

If `a` is a numpy array then `a + a` calculates

Selected Answer: A new array that sums each element of **a** with itself.

Answers:
- A string that is the concatenation of **a** with itself.
- Exactly the same as **a.append(a)**
- A new array that sums each element of **a** with itself.
- A new array that is the result of appending **a** to itself.

Response Feedback: You are right. The operator **+** applied to arrays calculates the sum element-wise.

Question 8

10 out of 10 points

If `a` is a list then `a * 2` computes

Selected Answer: A new list that is the result of cocatenating **a** to itself.

Answers:
- A new list where each element of **a** is multiplied by 2.

Exactly the same as **a.append(a)**

A new list that is the result of cocatenating **a** to itself.

A new list where each element of **a** is squared.

Response Feedback: You are right. The operator ***** applied to lists concatenates the two lists and creates a new list. This is not the same as **a.append(a)** because the latter modifies **a**.

Wednesday, 29 November 2023 09:18:12 o'clock GMT