

AI 310 Prerequisite Test *

Beaver-Edge AI Institute

If you are not sure whether you are ready to take AI 310, please complete this prerequisites test. There is no time limit on solving problems below.

If you can solve 70% of the problems, you are ready to take AI 310.

In this test, all problems are coding tasks. The only library you are allowed to import is NumPy.

1. You are given two NumPy arrays:

- (a) x with shape (n_0, \dots, n_{d-1}, N) where $d \geq 1$.
- (b) W with shape (M, N) .

Write one line of code with NumPy to compute matrix multiplication of these two arrays. The shape of the output array is (n_0, \dots, n_{d-1}, M) .

2. Do the following tasks:

- (a) Set the NumPy random seed as 42.
- (b) Generate NumPy arrays that are standard normal random variables with the following shapes:
 - i. $()$
 - ii. $(3,)$
 - iii. $(3, 5)$
 - iv. $(3, 5, 7)$
- (c) Release the NumPy random seed to its default value.

3. Use NumPy to generate 100 data points that are evenly spaced between -1 and 2, with these two endpoints that are inclusive.

4. Write code to generate the following NumPy array:

- (a) Shape: $(\text{num_samples}, \text{max_length})$ with $\text{num_samples}=20$ and $\text{max_length}=10$.
- (b) The length of each sample is uniformly distributed between 6 and 10 (both of these two numbers are inclusive).
- (c) If a sample has length k , then the first k numbers are normal random variables with mean k and standard deviation 2. The rest $\text{max_length}-k$ numbers are padded with 0.

*Copyright © Beaver-Edge AI Institute. All Rights Reserved. No part of this document may be copied or reproduced without the written permission of Beaver-Edge AI Institute.

Copyright © Beaver-Edge AI Institute. All Rights Reserved. No part of this document may be copied or reproduced without the written permission of Beaver-Edge AI Institute.

5. Write a function satisfying
- The input has two arguments. One is a 3-d NumPy array that stores a colorful image data. One is a positive integer k .
 - The output is a 3-d NumPy array that stores a colorful image obtained from the following transformation of the input data:
 - Do singular value decomposition on each color channel of the input image.
 - Keep top- k components in each channel.
 - While calling this function, display the output image.
6. Use NumPy to write a function that does the following tasks:
- Input: an arbitrary number of positional arguments `*args`.
 - The body of the function:
 - Create a NumPy array object that has shape `*args` and data type `uint8`.
 - Move axis 1 to the last position.
 - Flatten the array.
 - Normalize data within -1 and 1. Print and output it.
- To test your function, try the following inputs: `[1, 3, 4]`, `[3, 5, 2, 4]`.

7.

- (a) Write a function that uses NumPy to generate the following lattice points:

$$\{(x_0, x_1, \dots, x_{K-1}) \in \{0, 0.01, 0.02, \dots, 1\}^K\},$$

where the dimension K is the input.

The output is a NumPy array with shape `(101**K, K)`.

- (b) Set $K = 2$. Define

$$f(x_0, x_1) = \begin{cases} 3 & \text{if } \mathbf{1}\{x_0 + x_1 \leq 1\} \\ 1 & \text{elsewhere} \end{cases}.$$

Write code to generate this output. No loop is allowed.

- (c) For those points whose functional values are 3, plot them with the red color. For those points whose functional values are 1, plot them with the blue color.

8. Let

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(0),\top} \\ \mathbf{x}^{(1),\top} \\ \vdots \\ \mathbf{x}^{(N-1),\top} \end{bmatrix} \in \mathbb{R}^{N \times d}.$$

Write one line of code with NumPy to compute the following empirical covariance matrix

$$\frac{1}{N-1} \sum_{n=0}^{N-1} \hat{\mathbf{x}}^{(n)} \hat{\mathbf{x}}^{(n)\top},$$

where

$$\hat{\mathbf{x}}^{(n)} = \mathbf{x}^{(n)} - \frac{1}{N} \sum_{m=0}^{N-1} \mathbf{x}^{(m)}.$$

Loop is not allowed.

9.

- (a) Write one line of code with NumPy to generate an array with shape (N, N) , rank at most r , and each entry that is uniformly distributed between 0 and 1.
- (b) Use NumPy to do singular value decomposition of your generated 2-dim array.

10. You are given inputs μ with shape $(d,)$ and Σ with shape (d, d) and is guaranteed to be positive definite.

Write code with NumPy to generate a 2-dim array with shape (N, d) , denoted as `data`, such that `data[n]` for any $n = 0, \dots, N-1$ is a normal random variable with mean μ and covariance Σ .