VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**TRUONG PHUC NGUYEN – 520H0392**

**MITERM ESSAY**

**APPLIED LINEAR ALGEBRA FOR IT**

**HO CHI MINH CITY, 2024**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**TRUONG PHUC NGUYEN – 520H0392**

**MIDTERM ESSAY**

**APPLIED LINEAR ALGEBRA FOR IT**

Advised by

**Dr. Huynh Thi Thu Thuy**

**HO CHI MINH CITY, 2024**

**ACKNOWLEDGEMENT**

Dear Dr. Huynh Thi Thu Thuy

I wanted to take a moment to express my heartfelt gratitude for the incredible lecture you gave on "Applied Linear Algebra for IT". Your expertise, passion, and commitment to teaching were evident throughout the lecture, and I learned so much from you in a short period of time.

Once again, thank you for your exceptional lecture and for sharing your knowledge and expertise with us. Your contribution to my education and future success is deeply appreciated.

Thank you sincerely for everything.

*Ho Chi Minh city, 9th April 2024*

*Author*

*(Signature and full name)*

**

*Truong Phuc Nguyen*

**DECLARATION OF AUTHORSHIP**

We hereby declare that this is our own project and is guided by Dr. Huynh Thi Thu Thuy; The content research and results contained herein are central and have not been published in any form before. The data in the tables for analysis, comments and evaluation are collected by the main author from different sources, which are clearly stated in the reference section.

In addition, the project also uses some comments, assessments as well as data of other authors, other organizations with citations and annotated sources.

**If something wrong happens, We’ll take full responsibility for the content of my project.** Ton Duc Thang University is not related to the infringing rights, the copyrights that We give during the implementation process (if any).

*Ho Chi Minh city, 9th April 2024*

*Author*

*(Signature and full name)*

**

*Truong Phuc Nguyen*

**TABLE OF CONTENT**

[CHAPTER 1. METHODOLOGY FOR SOLVING TASKS 2](#_Toc163629252)

[1.1 Task 1d 2](#_Toc163629253)

[1.2 Task 1e 2](#_Toc163629254)

[1.3 Task 1f 2](#_Toc163629255)

[1.4 Task 1g 2](#_Toc163629256)

[1.5 Task 1h 3](#_Toc163629257)

[CHAPTER 2. SOURCE CODES AND OUTPUT 3](#_Toc163629258)

[2.1 Create 3 matrices 3](#_Toc163629259)

[2.2 Coding and Output 4](#_Toc163629260)

[2.2.1 Task 1a 4](#_Toc163629261)

[2.2.2 Task 1b 5](#_Toc163629262)

[2.2.3 Task 1c 6](#_Toc163629263)

[2.2.4 Task 1d 6](#_Toc163629264)

[2.2.5 Task 1e 7](#_Toc163629265)

[2.2.6 Task 1f 7](#_Toc163629266)

[2.2.7 Task 1g 8](#_Toc163629267)

[2.2.8 Task 1h 8](#_Toc163629268)

# METHODOLOGY FOR SOLVING TASKS

## Task 1d

Using python operators → **A[A%2==1]** will return odd values in matrix A.

## Task 1e

* Define an **isPrime()** function that will check if a number is prime.
* Create an empty vector called **task1e** to save the resultant vector.
* Conduct a for loop through matrix A, check each value by **isPrime()** then append to **task1e** if true.

## Task 1f

* Calculate matrix D by using **numpy.dot(C,B)**
* Create a **task1f** matrix that copy matrix D
* Select odd rows: **task1f[::2]**
* Then reverse row values by **task1f[::2,::-1]**

## Task 1g

* Already have **isPrime()** function in task 1e
* Define a **maxPrimeRow()** to do the requirement:
  + Create an empty **temp** vector to save the resultant
  + Iterate each row to count the number of prime number

→ get the **maxCount** of prime number in matrix A

* When iterate over the rows, if find a row have the **maxCount** > current **maxCount**

→ Update **maxCount** value and store the maxCount row to **temp**

## Task 1h

* Define a **longestOddSeq()** to do the requirement
  + Create an empty vector **temp** to save the resultant.
  + Initialize **countOdd** value to save the value of maximum contiguous odd number.
  + Conduct a for loop to check if an element is odd number

→ **countOdd** +1

* + If an element is even number

→ reset **countOdd**

* + If the current count value > **countOdd**

→ Store the current row to **temp**.

# SOURCE CODES AND OUTPUT

## Create 3 matrices

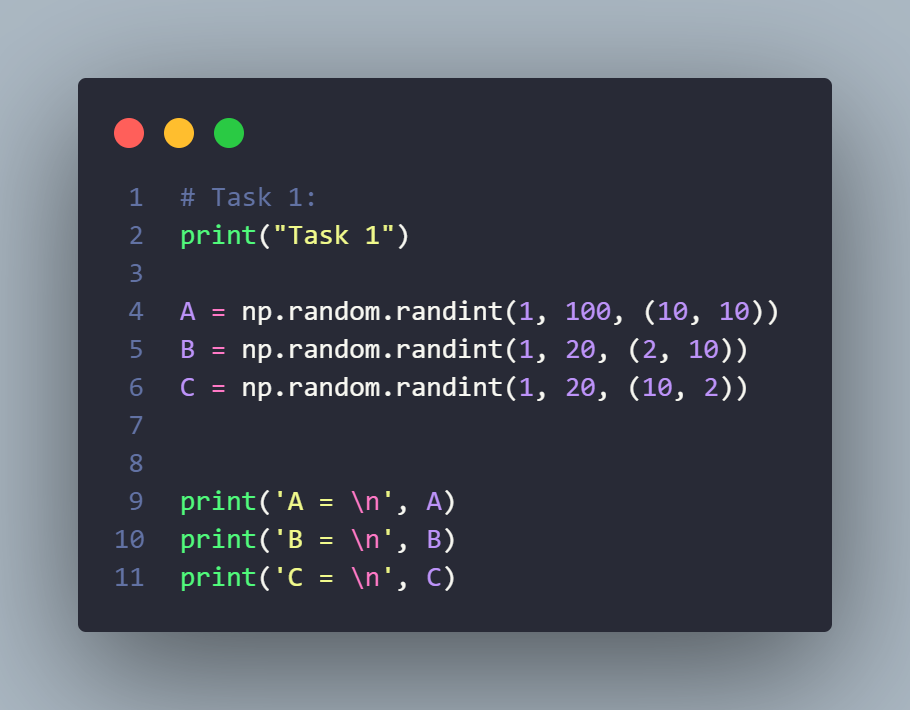
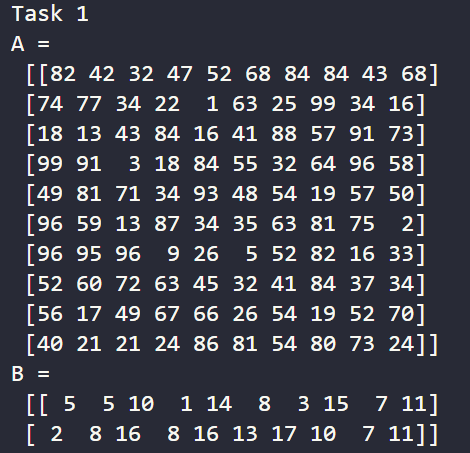
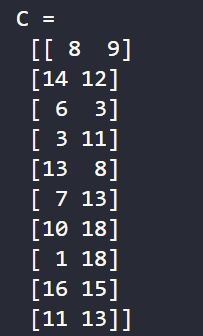


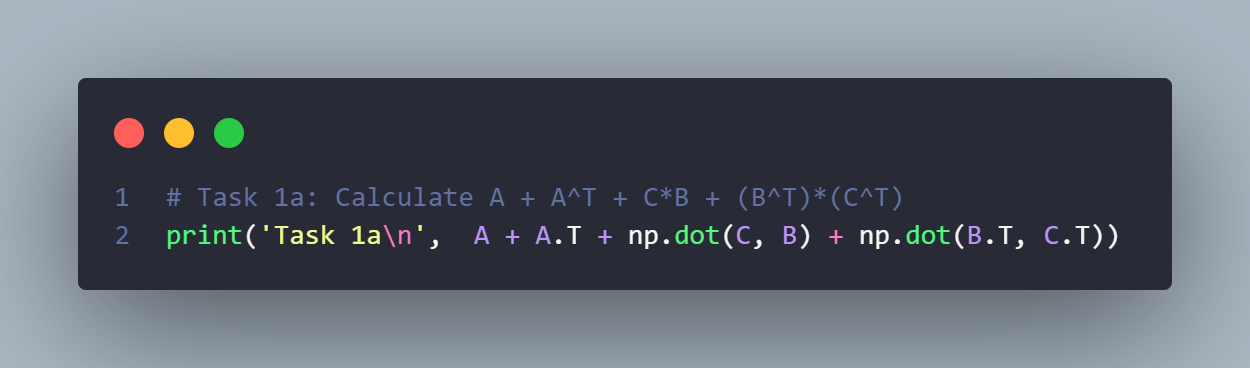
Figure Create matrices A, B, C

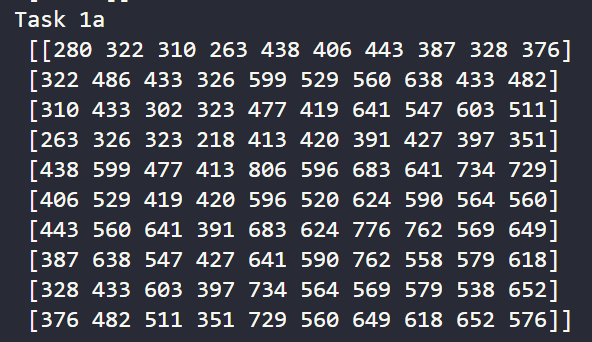
Ouput 3 matrices

## Coding and Output

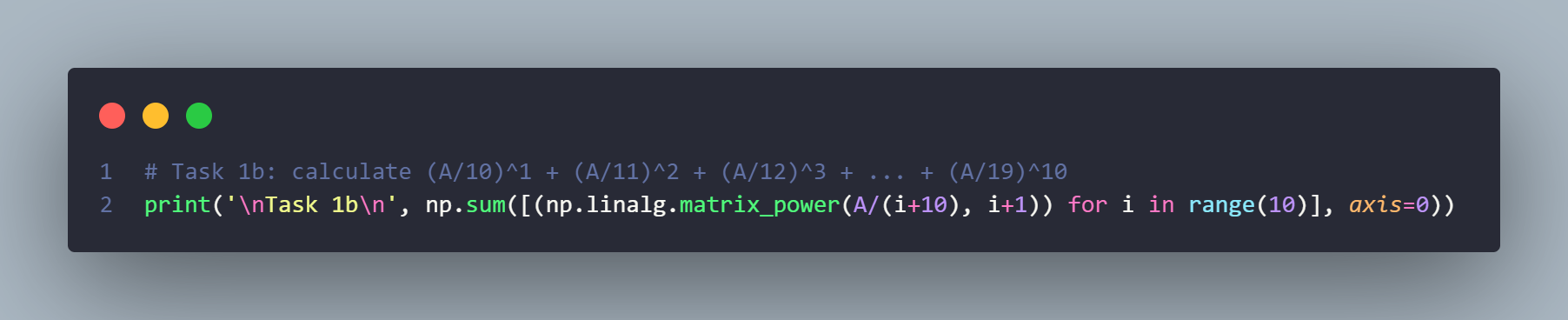
### Task 1a



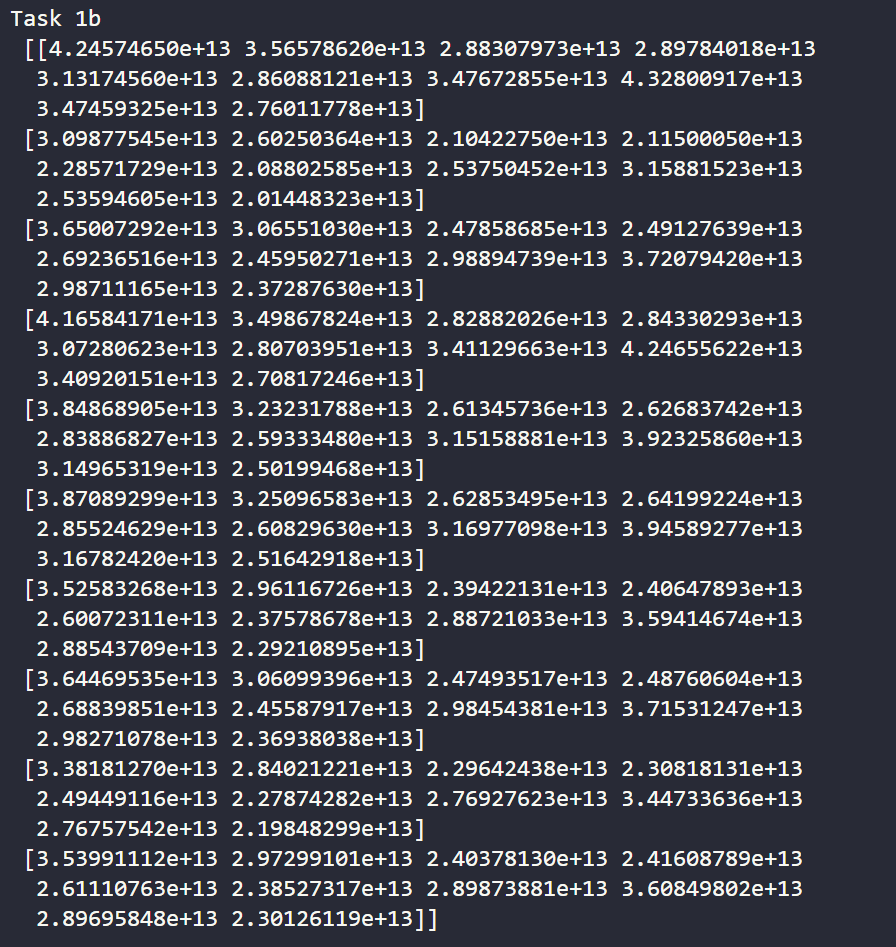
Output:



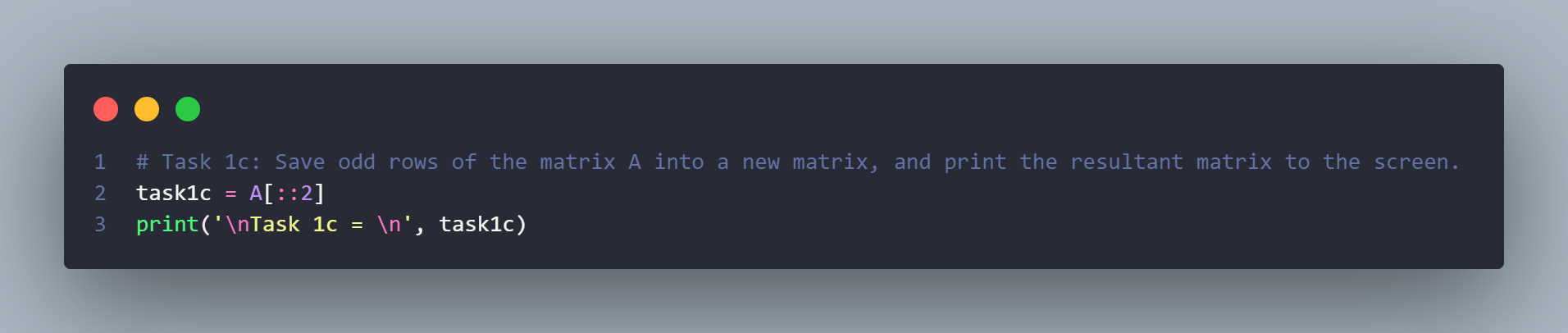
### Task 1b



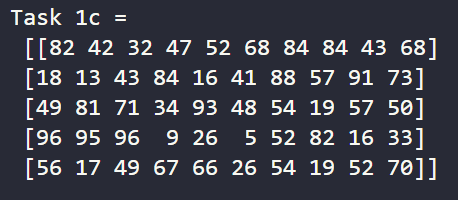
Output:



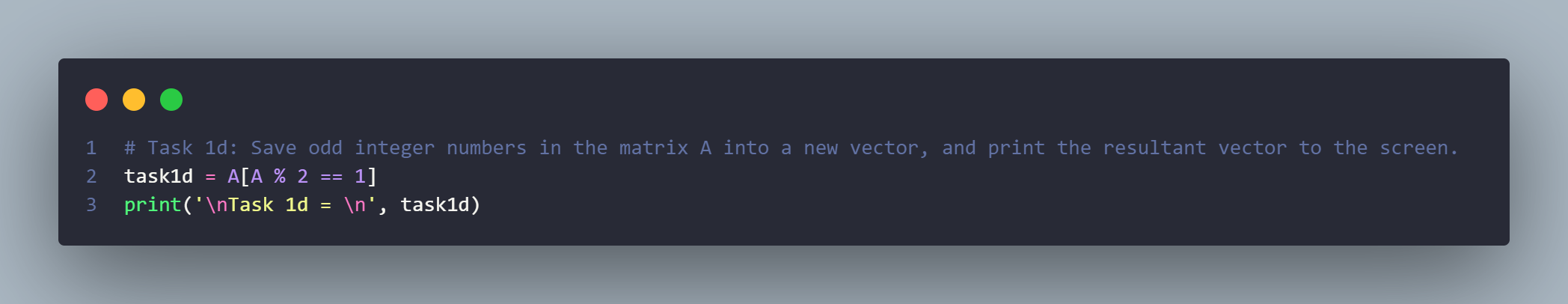
### Task 1c



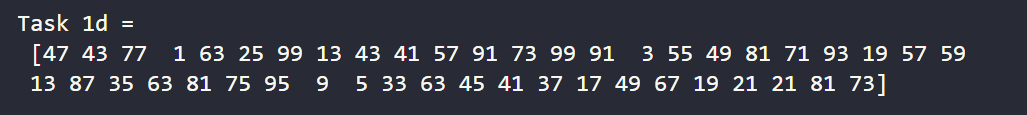
Output:



### Task 1d



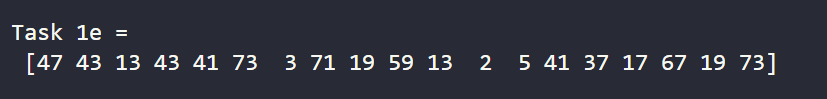
Output:



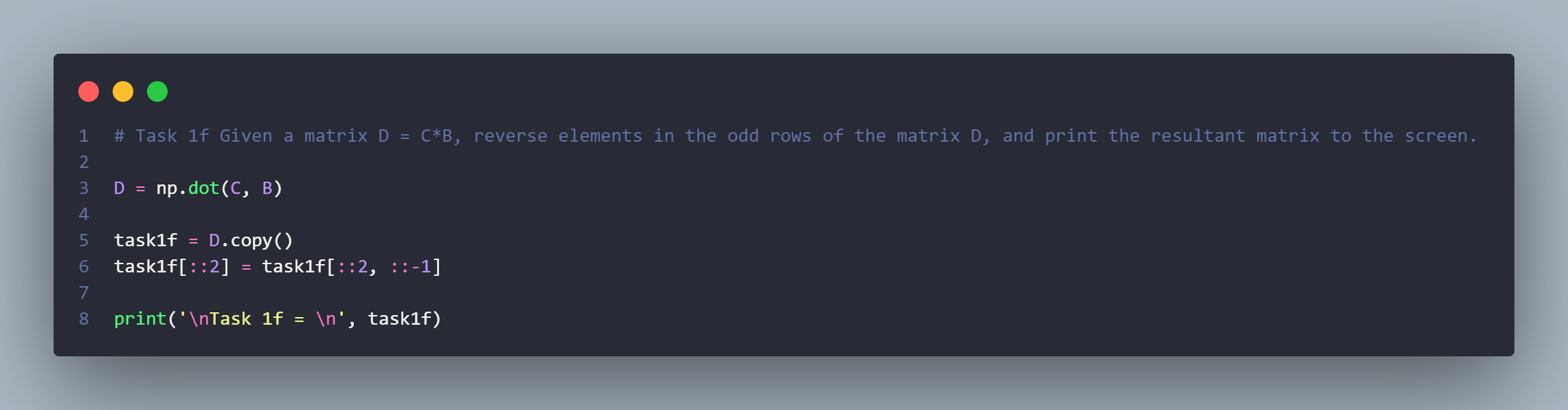
### Task 1e



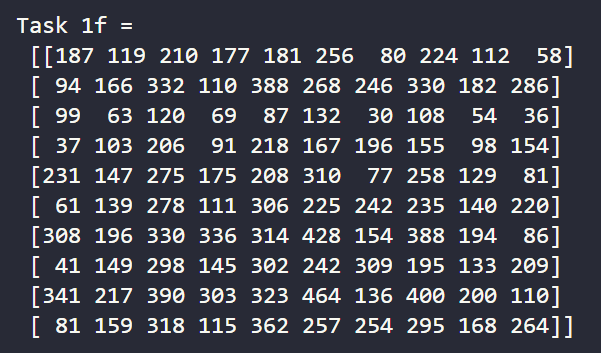
Output:



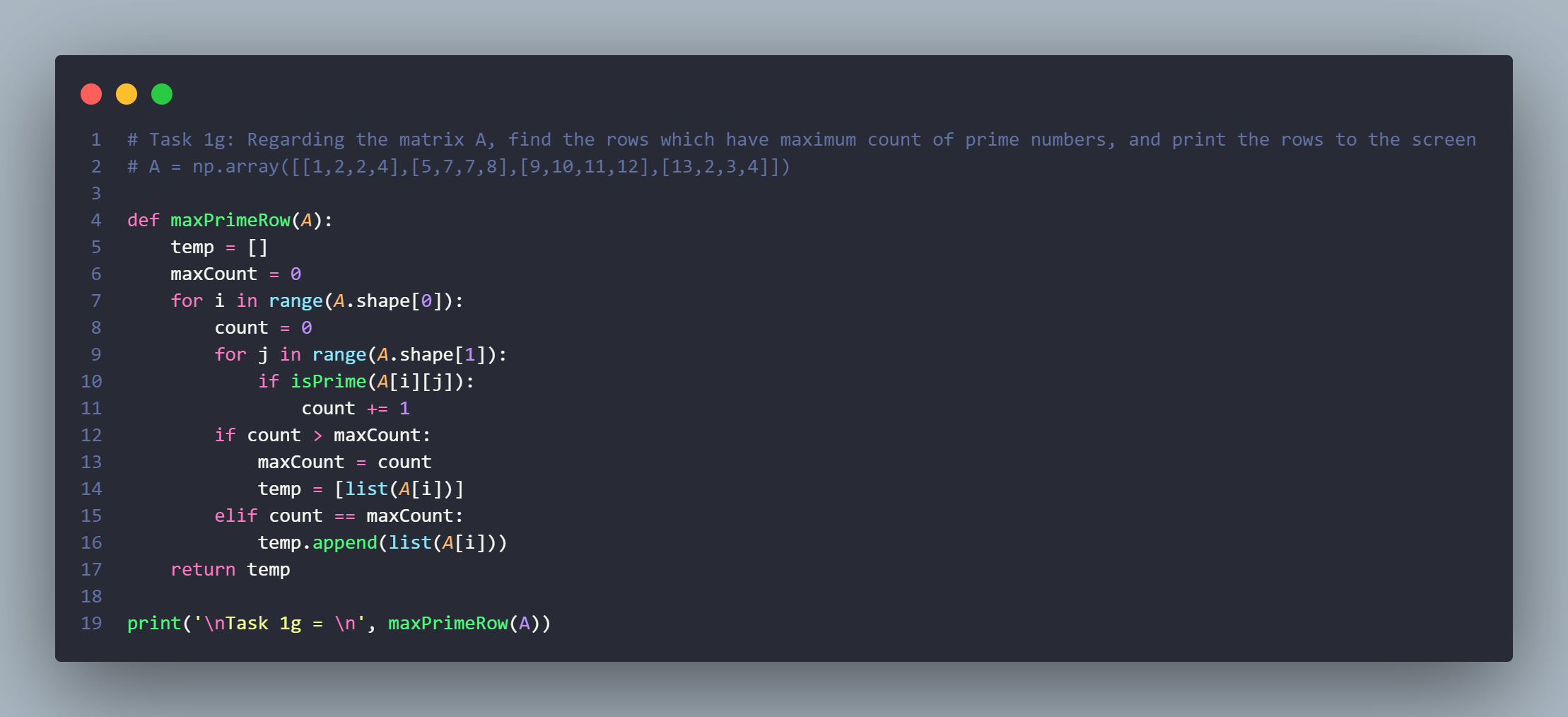
### Task 1f



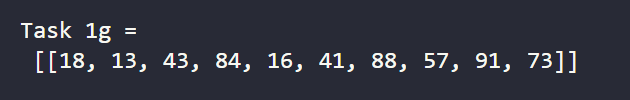
Output:



### Task 1g



Output:



### Task 1h



Output:

