Numerical Method

National Cheng Kung University Department of Engineering Science

Instructor: Chi-Hua Yu

Name:_____

Student ID:_____

Midterm Mock Exam

注意事項

1. **本次模擬考試不需繳交。**
2. 程式題部分，請繳交 **ipynb 的檔案形式**，並輸入正確的檔名。
3. 請用**學號_Midterm** 為檔名做一個資料夾**(e.g., N96091350_Midterm)**，並將程式題之**.ipynb 檔案**放入資料夾中，壓縮後上傳至課程網站**(e.g., N96091350_Midterm.zip)**。
4. 如未依照上述規則繳交作業、繳交錯誤檔案，則以零分計算，不允許要分。
5. 手寫題可跳題作答，但必須標示清楚題號，若題號標示錯，該題也會視為零分，不允許要分。如字跡潦草至助教難以辨別，則會以助教辨視為主。
6. 程式題請依照題目規定作答，若無依照題目則將該題視為零分，不允許要分。
7. 請注意作答時不要抄襲網路或是同學的答案，助教會將程式碼放入自動比對程式，只要超過 70%相似度，以抄襲處置，抄襲者與被抄襲者都以零分計算。
8. 本次閱卷將採用自動批改，命名錯誤或是無法執行將被自動判定為失敗，失去該題的分數。成功通過自動閱卷的程式碼，助教會再進行人工判讀，確定程式邏輯是否恰當，因此添加最低限度的註解可以保障作答時候的分數。

請勿抄襲，抄襲者與被抄襲者本次考試皆 0 分計算

Numerical Method

National Cheng Kung University Department of Engineering Science
Instructor: Chi-Hua Yu

Total (120%)

Part I (15%) Conceptual Problems: answer the following questions **briefly.**

1. **(10%) (a) (5%)** What is the local truncation error (error complexity) of trapezoid rule approximation integral method?

   (b) (5%) What is the local truncation error (error complexity) of Simpson's rule approximation integral method?

2. **(5%) (a) (2%)** What is data type of `a = [1, 3, 5 ,7]`

   (b) (3%) What is data type of `a = np.array([1, 3, 5 ,7])`

Part II (35%) Derivation Problems: give detailed derivations of the following questions.

1. **(35%)** Use the power method to obtain the largest eigenvalue and eigenvector for the matrix A= $\begin{bmatrix} 2 & 1 & 2 \\ 1 & 3 & 2 \\ 2 & 4 & 1 \end{bmatrix}$ Start with initial vector $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ and see the results after three iterations.

Part III (70%) Programming Problems.

1. **(35%)** Name your Jupyter notebook `Gauss_Elimination.ipynb`. Write a Python program to solve the equations by using the Gauss elimination method. Use the Gauss elimination method to solve the following equations:

$$-2x_2 + 6x_3 = 7,$$
$$12x_1 + 4x_2 - 7x_3 = 5,$$
$$4x_2 + x_3 = 3.$$

   Below is the running example:

   (The numeric precision of the result is to three decimal places.)

```
a = np.array([[0,-2,6],[12,4,-7],[0,4,1]]).astype('float')
y = np.array([7,5,3]).astype('float')
x = Gauss_Elimination(a, y)
print('x:', x)


x: [1.03846154 0.42307692 1.30769231]
```

2. **(35%)** Name your Jupyter notebook `Power_Method.ipynb`. Write a Python program to find the largest eigenvalue and the associated eigenvector by using the power method. You can use [1, 1] as the initial vector x to start the iteration. In each iteration is usually normalized, which will make the largest element in the vector equal to 1. Normalization will provide the largest eigenvalue and its corresponding eigenvector at the same time. Use the Power method to find the largest eigenvalue and the associated eigenvector of following equations:

$$a = \begin{bmatrix} 2 & 3 \\ 3 & -1 \end{bmatrix}$$

   Below is the running example:

   (The numeric precision of the result is to three decimal places.)

```
x = np.array([1, 1])
a = np.array([[2, 3],[3, -1]])
Eigenvalue, Eigenvector = Power_Method(a, x)
print("Eigenvalue:",  Eigenvalue)
print("Eigenvector:", Eigenvector)


Eigenvalue: 3.8541019662496843
Eigenvector: [1.         0.61803399]
```