الجلسة العملية السادسة

عنوان الجلسة: ايجاد جذور التوابع -القسم الثاني

الغاية من الجلسة :

متابعة دراسة خوارزميات ايجاد جذور التوابع بالطرق العددية و تحويلها الى أكواد برمجية بلغة البايثون

Fixed Point Algorithm

3-خوارزمية النقطة الثابته

- طريقةالنقطة الثابته: تختلف هذه الطريقّة عن الطرق السابقة بأنها تبدأ من الدالة $f(x)=0$ للحصول على الشكل $g(x)-x=0$ ثم نستخدم هذه المعادلة بالحصول على الدستور التكراري $x(i+1)=g(xi)$ لإيجٌاد المتتاليه $x0,x1,x2,.....$ انطلاقا من التقر يبٌ الابتدائي لها

الخوارزمية:

إذا كانت الدالةلدينا الدالة $f(x)=0$ قابلة للكتابه بالشكل: $x=g(x)$

1- نعرف مدخلات الخوارزمية: $x0$النقطة الابتدائية, tol سماحية الخطأ, التابع $f(x)$ عدد مرات التكرار , قيمة ابتدائية لعداد حلقة التكرار.

2- نكرر تنفيذ الخطوات التالية:

2-1 نوجد $x1=g(x0)$

2-2 إذا كانت قيمة $|x1-x0|<tol$ فإن $x1$جذر للدالة $f(x)$ وحلا لها و بالتالي نطبع قيمة الجذر و نوقف الحلقة.

2-3 نضع $x0 = x1$ و نزيد عداد الحلقة

3- نطبع العبارة لا يوجد حل.

التمرين الأول:

بما أن الخوارزمية هي خوارزمية تكرارية يمكن حل التمرين باستخدام الحلقات او باستخدام التوابع العودية أو باستخدام تابع بداخله حلقة

لا حظ أن نص التمرين يطلب الحل بالحلقات

في بايثون :

1- استخدم الحلقات لايجاد الجذر التقريبي لتابع معطى $f(x)=x-e^{(-x)}=0$ مستخدما خوارزمية النقطة الثابته.

من أجل نقطة البداية , $x0=0$, ومن أجل 10 تكرارات , بدقة $tol=1E-6$, اطبع على الشاشة قيمةالجذر و قيمة االتابع f عند هذا الجذر اذا وجد , او اطبع العبارة لا يوجد حل عند عدم وجود حل.

2-اخرج نتائجك في جدول.

الأدوات اللازمة لحل التمرين:

1-أدوات الشرط في python (تعرفنا إليه سابقا)

3-استخدام Lambda لتعريف التابع المطلوب دراسته .(تعرفنا إليه سابقا)

4-استخدام حلقة while

5- استخدام مكتبة tabulate

الأدوات اللازمة لحل التمرين:

استخدام while في python:

تبقى الحلقة في حالة التنفيذ طالما
أن الشرط يأخذ القيمة True

Python while Loop

```
while condition:
     # body of while loop
```

في Python ، قد تحتوي حلقة while على كتلة else اختيارية.

هنا ، يتم تنفيذ الجزء الآخر بعد تقييم حالة الحلقة إلى False.

ملاحظة: لن يتم تنفيذ كتلة else إذا تم إنهاء حلقة while بواسطة تعليمة break

Python While loop with else

```
while condition:
    # body of while loop
else:
    # body of else
```

الأدوات اللازمة لحل التمرين:

استخدام while في python:

Python break and continue

```
while condition:
  # code
  if condition:
    break

  # code
```

```
while condition:
  # code
  if condition:
    continue

  # code
```

الأدوات اللازمة لحل التمرين:

استخدام tabulate في python:

وحدة الجدولة لإنشاء جداول في بايثون

```
from tabulate import tabulate  #استيراد وحدة الجدولة


print(tabulate(all_data, headers='firstrow',tablefmt='grid'))
```

شكل الجدول

ترويسة الجدول

قائمة تمثل اسطر الجدول كل عنصر في هذه القائمة هو قائمة

الكود البرمجي المتعلق بالحل باستخدام الحلقة:

```
import numpy as np
import math
from tabulate import tabulate

tol=1E-6
x0=0
i=1
n=10
f=lambda x:x-math.exp(-x)
header=["i","xi","f(xi)","|eps
ilon|,%"]
#
epsilon=np.abs((x1-x0)/x1)*100
mydata=[[0,0,f(0),       ]]
```

اول عنصر في القائمة
او اول سطر في الجدول

ترويسة الجدول

```
while i<=n:
  x1=math.exp(-x0)
  if np.abs(x1-x0)<tol:
    print("x1=",x1)
    k=f(x1)
    print(k)
    break
  epsilon=np.abs((x1-x0)/x1)*100
  mydata.append([i,x1,f(x1),epsilon])
  x0=x1
  i=i+1
else :
  print("There Is No Solution")
print(tabulate(mydata, headers=header,
tablefmt="grid"))
```

OUTPUT

There Is No Solution

i	xi	f(xi)	|epsilon|,%
0	0	-1	
1	1	0.632121	100
2	0.367879	-0.324321	171.828
3	0.692201	0.191727	46.8536
4	0.500474	-0.10577	38.3091
5	0.606244	0.0608477	17.4468
6	0.545396	-0.0342165	11.1566
7	0.579612	0.0194969	5.90335
8	0.560115	-0.0110277	3.48087
9	0.571143	0.00626377	1.9308
10	0.564879	-0.00354938	1.10887

التمرين الثاني

بما أن الخوارزمية هي خوارزمية تكرارية يمكن حل التمرين باستخدام الحلقات او باستخدام التوابع العودية أو باستخدام تابع بداخله حلقة

لاحظ أن نص التمرين يطلب الحل بشكل تابع

استخدم بايثون:

1- لكتابة تابع لايجاد الجذر التقريبي لتابع معطى f(x)=x-e^(-x)=0 مستخدما خوارزمية النقطة الثابته

من أجل نقطة البداية , x0=0 ,ومن أجل 10 تكرارات , بدقة tol=1E-6 ,اطبع على الشاشة قيمةالجذر و قيمة االتابع f عند هذا الجذر اذا وجد , او اطبع العبارة لا يوجد حل عند عدم وجود حل.

2- اخرج نتائجك في جدول

الأدوات اللازمة لحل التمرين:

1-كتابة تابع. (تعرفنا إليه سابقا)

2-أدوات الشرط في python (تعرفنا إليه سابقا)

3-استخدام Lambda لتعريف التابع المطلوب دراسته .(تعرفنا إليه سابقا)

4-استخدام حلقة while

5- استخدام مكتبة tabulate

```python
import numpy as np
import math
from tabulate import tabulate
def My_FxixedPoint(x0,tol,f,n):
  i=1
  while i<=n:
    x1=math.exp(-x0)
    if np.abs(x1-x0)<tol:
      print("x1=",x1)
      k=f(x1)
      print(k)
      break
    epsilon=np.abs((x1-x0)/x1)*100
    mydata.append([i,x1,f(x1),epsilon])
    x0=x1
    i=i+1
  else :
    print("There Is No Solution" )
```

الكود البرمجي المتعلق بالحل باستخدام التابع:

هنا استخدمنا تابع بداخله حلقة

```python
tol=1E-6
x0=0
n=10
f=lambda x:x-math.exp(-x)
header=["i","xi","f(xi)","|epsilon|,%"]#
epsilon=np.abs((x1-x0)/x1)*100
mydata=[[0,0,f(0),      ]]
My_FxixedPoint(x0,tol,f,n)
print(tabulate(mydata, headers=header,
tablefmt="grid"))
```

OUTPUT

There Is No Solution

i	xi	f(xi)	|epsilon|,%
0	0	-1	
1	1	0.632121	100
2	0.367879	-0.324321	171.828
3	0.692201	0.191727	46.8536
4	0.500474	-0.10577	38.3091
5	0.606244	0.0608477	17.4468
6	0.545396	-0.0342165	11.1566
7	0.579612	0.0194969	5.90335
8	0.560115	-0.0110277	3.48087
9	0.571143	0.00626377	1.9308
10	0.564879	-0.00354938	1.10887

Secant Algorithm

4-خوارزمية القاطع أو لاغرانج

طريقة القاطع هي طريقة مفتوحة وتبدأ بتخمينين أوليين لإيجاد الجذر الحقيقي للمعادلات غير الخطية.

في طريقة secant إذا كانت x0 و x1 عبارة عن تخمينات أولية ، فسيتم الحصول على الجذر التقريبي التالي x2 بالصيغة التالية:

```
x2 = x1 - (x1-x0) * f(x1) / ( f(x1) - f(x0) )
```

وتتضمن خوارزمية طريقة Secant تكرار العملية المذكورة أعلاه ، أي نستخدم x1 و x2 لإيجاد x3 وما إلى ذلك حتى نجد الجذر ضمن الدقة المطلوبة.

الخوارزمية:

```
1. Start

2. Define function as f(x)

3. Input initial guesses (x0 and x1),
   tolerable error (e) and maximum iteration (N)

4. Initialize iteration counter i = 1

5. If f(x0) = f(x1) then print "Mathematical Error" and goto (11) otherwise goto (6)

6. Calcualte x2 = x1 - (x1-x0) * f(x1) / ( f(x1) - f(x0) )

7. Increment iteration counter i = i + 1

8. If i >= N then print "Not Convergent"
   and goto (11) otherwise goto (9)

9. If |f(x2)| > e then set x0 = x1, x1 = x2
   and goto (5) otherwise goto (10)

10. Print root as x2

11. Stop
```

التمرين الثالث :

بما أن الخوارزمية هي خوارزمية تكرارية يمكن حل التمرين باستخدام الحلقات او باستخدام التوابع العودية

لا حظ أن نص التمرين لا يطلب الحل بشكل محدد لكننا هنا سنستخدم الحلقة

استخدم بايثون:

1- لايجاد الجذر التقريبي لتابع معطى مستخدما خوارزمية **القاطع**

استخدم ما سبق لايجاد الجذر التقريبي للمعادلة `f(x)=e^x+3*x` من أجل الحل الابتدائي $x1=1$ $x0=-1$ من أجل 10 تكرارات و بدقة 1E-4. و اطبع على الشاشة قيمة التابع f عند هذا الجذر .

ملاحظة : القيمة 1E-4 تعني 0.0001 او (10^-4)

الكود البرمجي

```
import numpy as np
from tabulate import tabulate

f=lambda x: np.exp(2*x)+3*x
x0=-1;x1=1
i=0
n=10

header=["i","x0","x1","f(x1)","f(x1)<Epsilon"]
my_secant_result=[[0,-1,1,f(1),f(1)<1E-4]]
```

```
while i<=n :
  i=i+1
  x2=x1-(f(x1)*(x1-x0)/(f(x1)-f(x0)))
  if np.abs(f(x1))<=1E-4:
    print(x2)
    k=f(x2)
    print(k)
    break
  x0=x1
  x1=x2

my_secant_result.append([i,x0,x1,f(x1),np. ab
s(f(x1))<1E-4])
else:
  print("No Solution For This Function" )
print(tabulate(my_secant_result,
headers=header, tablefmt= "grid"))
```