ASSIGNMENT 3: **NUMERICAL COMPUTING THROUGH PYTHON 3.9.2**

Section 4G

Group Members:

Muhammad Ibad Saleem	(19K0220)
Ali Hamza Usmani	(19K0146)
Shah Tanzeel Ahmed	(19K0161)
Abdul Rehman Ahmed	(19K0166)

Contents

Introduction

Numerical computing is the vast field in the computer science that deals with the computations and arithmetic problems in the computing world. Numerical Computing deals with solving complex and derived based equations through simple derived formula and mainly computational iterations.

Objectives

The main objective to perform coding for the numerical computing methods are to speed up the process of finding the accurate roots of the equation as in most of the cases the iterations can go up to 25 cycles and beyond as per the complexity and the nature of the equation.

Why Python?

We choose Python due to the fact that in today's world Python holds an esteem value of being a go to programming language in each and every field of computer science such as Data Science, Artificial Intelligence, Deep Learning and Statistical Computations etc. Numerical Computing through Python makes it easier for the beginners to understand how computations are performs due to the fact that python is one of the simplest language with simple semantics and syntax.

Methods Performed Of Numerical Computing

Following are some of the methods performed through Python programming language.

Chapter 2: *Solutions of Equations in One Variable Methods*

Chapter 3: *Interpolation and Polynomial Approximation Methods*

Chapter 4: *Numerical Differentiation and Integration Methods*

Chapter 5: *Initial-Value Problems for Ordinary Differential Equations Methods*

CHAPTER 2	CHAPTER 3	CHAPTER 4	CHAPTER 5
Bisection Method	Lagrange Interpolation	Three Point End/Mid Point Difference	Euler Method
Regular Falsi/False Position	Newton Divided Difference	Five Point End/Mid Point Difference	Modified Euler ODE
Secant Method	Newton Forward Difference	Trapezoidal Integration	MidPoint ODE
Collective Analysis	Newton Backward Difference	Simpson $\frac{1}{3}rd$ and $\frac{3}{8}th$ Integration	*ODE => Ordinary Differential Equation

Code Sample Outputs (3 Methods Output Shown)

OUTPUT	DESCRIPTION
NUMNERICAL COMPUTING IN PYTHON IBAD SALEEM TANZEEL AHMED ABDUL REHMAN ALI HAMZA USMANI PRESS ANY KEY CONTINUE =	**Opening Screen Of Program with Introduction**

```
1. Chapter 2: Solutions of Equations in One Variable Methods
2. Chapter 3: Interpolation and Polynomial Approximation Methods
3. Chapter 4: Numerical Differentiation and Integration Methods
4. Chapter 5: Initial-Value Problems for Ordinary Differential Equations Methods
>>>
```

Chapter Menu To be chosen for performing computation

C:\Windows\SYSTEM32\cmd.exe

```
1. Chapter 2 methods
2. Chapter 3 methods
3. Chapter 4 methods
4. chapter 5 methods
>>> 1
1. bisection
2. Regular Falsi
3. secant method
4. Collective analysis
>>> 1
Enter the variable : x
Enter equation with the proper syntax :  x^3 +(4*x^2) -10
enter value of a=1
enter value of b=2
enter tolerance value [t] 10^-[t] = (enter only t after minus sign) = 3
2.375
                a               b               c            f(a)            f(b)            f(c)       abs.error

              1.0             2.0             1.5            -5.0            14.0           2.375           2.375
              1.0             1.5            1.25            -5.0           2.375         -1.7969          4.1719
             1.25             1.5           1.375          -1.797           2.375          0.1621           1.959
             1.25           1.375          1.3125          -1.797           0.162         -0.8484          1.0105
           1.3125           1.375          1.3438          -0.848           0.162         -0.3502          0.4982
           1.3438           1.375          1.3594           -0.35           0.162          -0.096          0.2542
           1.3594           1.375          1.3672          -0.096           0.162          0.0326          0.1286
           1.3594          1.3672          1.3633          -0.096           0.033         -0.0318          0.0644
           1.3633          1.3672          1.3653          -0.032           0.033          0.0012           0.033
           1.3633          1.3653          1.3643          -0.032           0.001         -0.0154          0.0166
           1.3643          1.3653          1.3648          -0.015           0.001         -0.0071          0.0083
           1.3648          1.3653          1.3651          -0.007           0.001         -0.0021           0.005
           1.3651          1.3653          1.3652          -0.002           0.001         -0.0005          0.0016
           1.3652          1.3653          1.3653            -0.0           0.001          0.0012          0.0017
number of iterations = 14

Solution = 1.3653
```

Chapter 2 Bisection Method Implemented.

C:\Windows\SYSTEM32\cmd.exe

```
4. chapter 5 methods
>>> 1
1. bisection
2. Regular Falsi
3. secant method
4. Collective analysis
>>> 2
Enter the variable : x
Enter equation with the proper syntax :  cos(x)-x
cos(x)-x
enter value of a=0.5
enter value of b=pi/4
enter tolerance value [t] 10^-[t] = (enter only t after minus sign) = 6
                a                    b                    c               f(a)               f(b)               f(c)          abs.error

              0.5   0.7853981633974483            0.7363841           0.377583          -0.078291          0.0045178          0.0045178
        0.7363841   0.7853981633974483            0.7390581           0.004518          -0.078291           4.52e-05          0.0044726
        0.7390581   0.7853981633974483            0.7390849            4.5e-05          -0.078291              4e-07           4.48e-05
number of iterations = 3

Solution = 0.7390849
1. Chapter 2 methods
2. Chapter 3 methods
3. Chapter 4 methods
4. chapter 5 methods
>>>
```

Chapter 2 Regular Falsi Method Implemented

```
C:\Windows\SYSTEM32\cmd.exe
1. Chapter 2 methods
2. Chapter 3 methods
3. Chapter 4 methods
4. chapter 5 methods
>>> 1
1. bisection
2. Regular Falsi
3. secant method
4. Collective analysis
>>> 3
Enter the variable : x
Enter equation with the proper syntax :  sin(x)-e^(x*-1)
sin(x)-e^(x*-1)
enter value of a=0
enter value of b=1
enter tolerance value [t] 10^-[t] = (enter only t after minus sign) = 5
                   a                   b                   c                f(a)                f(b)                f(c)           abs.error

                 0.0                 1.0            0.678614                -1.0             0.47359            0.120395            0.120395
                 1.0            0.678614            0.569062             0.47359              0.1204           -0.027214            0.147609
            0.678614            0.569062             0.58926              0.1204            -0.02721            0.001008            0.028222
            0.569062             0.58926            0.588538            -0.02721             0.00101               7e-06            0.001001
number of iterations = 4

Solution = 0.588538
1. Chapter 2 methods
2. Chapter 3 methods
3. Chapter 4 methods
4. chapter 5 methods
>>>
```

Chapter 2 Secant Method Implemented

```
>>> 2
1. Lagrange Interpolation
2. Newton Divided Difference
3. Newton Forward Difference Formulae
4. Newton Backward Difference Formulae
>>> 1
prom = 1
1)from table    2) from equation
2
Enter the variable : x
Enter equation with the proper syntax :  1/x
emter number of digits = 3
x0=2
x1=2.75
x2=4
{2.0: 0.5, 2.75: 0.36363636363636365, 4.0: 0.25}
value at  = 3
3
                                        for degree 1
x = 3.0  x0 = 2.75   x1/x2 = 4.0
0.3636363636

x = 3.0  x0 = 4.0   x1/x2 = 2.75
0.25


solution = 0.34090909088
```

Chapter 3 Lagrange Interpolation Implemented

```
4. Newton Backward Difference Formulae
>>> 2
prom = 2
emter number of entries = 5
x0=1
y0=0.7651977
x1=1.3
y1=0.6200860
x2=1.6
y2=0.4554022
x3=1.9
y3=0.2818186
x4=2.2
y4=0.1103623
value at  = 1.5




[1.0, 1.3, 1.6, 1.9, 2.2]
[0.7651977, 0.620086, 0.4554022, 0.2818186, 0.1103623]
[-0.4837056667, -0.548946, -0.578612, -0.571521]
[-0.1087338888, -0.0494433333, 0.0118183333]
[0.065878395, 0.0680685184]
[0.0018251028]
0.5118126938
```

Chapter 3 Newton Backward Difference Implemented

```
4. Chapter 5: Initial-Value Problems for Ordinary Differential Equations Methods
>>> 2
1. Lagrange Interpolation
2. Newton Divided Difference
3. Newton Forward Difference Formulae
4. Newton Backward Difference Formulae
>>> 3
prom = 3
emter number of entries = 3
x0=-0.1
y0=5.3
x1=0.0
y1=2
x2=0.2
y2=3.19
value at  = 0.15


[-0.1, 0.0, 0.2]
[5.3, 2.0, 3.19]
[-3.3, 1.19]
[4.49]
5.46875
```

Chapter 3 Newton Forward Difference Implemented

Select C:\WINDOWS\SYSTEM32\cmd.exe

```
1. Chapter 2: Solutions of Equations in One Variable Methods
2. Chapter 3: Interpolation and Polynomial Approximation Methods
3. Chapter 4: Numerical Differentiation and Integration Methods
4. Chapter 5: Initial-Value Problems for Ordinary Differential Equations Methods
>>> 3
1. Three Point End Point Differentiation
2. Three Point Mid Point Differentiation
3. Five Point Endpoint Differentiation
4. Five Point Mid Point Differentiation
5. Trapezonial Integration
6. Simpson Integration (1/3)rd
7. Simpson Integration (3/8)th
>>> 1
prom = 1
Enter Variables: x
Enter Equation: x*e^(x)
x0 = 2
h = 0.1
f`(x) = 22.032304865499963
true error0.1348634
```

Chapter 4 Three Point Endpoint Differentiation Implemented.

C:\WINDOWS\SYSTEM32\cmd.exe

```
4. Chapter 5: Initial-Value Problems for Ordinary Differential Equations Methods
>>> 3
1. Three Point End Point Differentiation
2. Three Point Mid Point Differentiation
3. Five Point Endpoint Differentiation
4. Five Point Mid Point Differentiation
5. Trapezonial Integration
6. Simpson Integration (1/3)rd
7. Simpson Integration (3/8)th
>>> 2
prom = 2
Same equation?Y
x0 = 2
h = 0.1
f`(x) = 22.22878688049999
true error0.0616186
1. Chapter 2: Solutions of Equations in One Variable Methods
2. Chapter 3: Interpolation and Polynomial Approximation Methods
3. Chapter 4: Numerical Differentiation and Integration Methods
4. Chapter 5: Initial-Value Problems for Ordinary Differential Equations Methods
>>>
```

Chapter 4 Three Point Midpoint Differentiation Implemented.

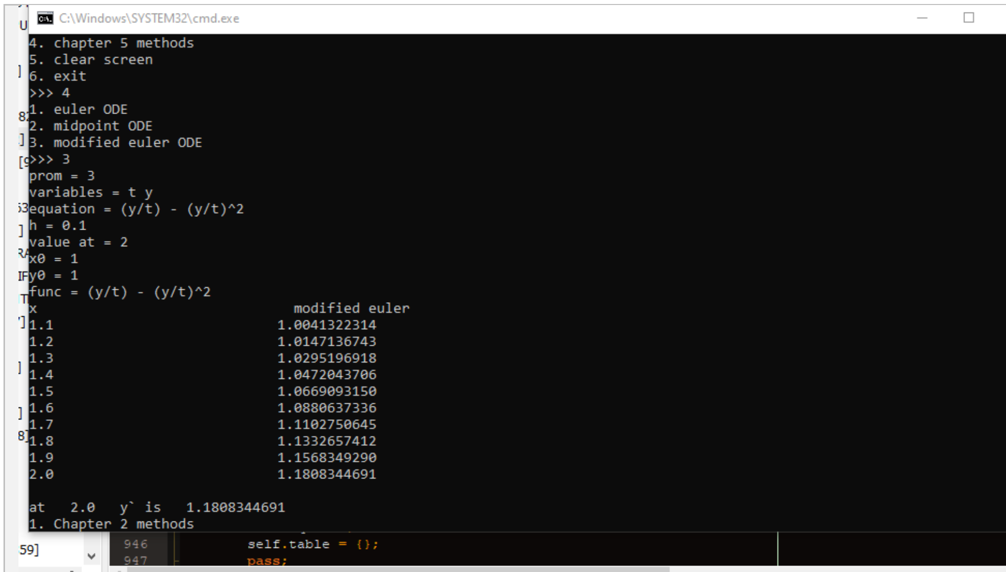
```
1. Chapter 2: Solutions of Equations in One Variable Methods
2. Chapter 3: Interpolation and Polynomial Approximation Methods
3. Chapter 4: Numerical Differentiation and Integration Methods
4. Chapter 5: Initial-Value Problems for Ordinary Differential Equations Methods
>>> 3
1. Three Point End Point Differentiation
2. Three Point Mid Point Differentiation
3. Five Point Endpoint Differentiation
4. Five Point Mid Point Differentiation
5. Trapezonial Integration
6. Simpson Integration (1/3)rd
7. Simpson Integration (3/8)th
>>> 5
prom = 5
Enter Variables: x
enter equation = x^2
enter lower limit a = 0
enter upper  limit b = 2
enter value of n2
intg(f(x)) = 3.0
lower = 0.0   upper = 2.0
func = x**2
acc = 2.66666666666667
true error = 0.3333333
```

Chapter 4 Five Point Endpoint Differentiation Implemented.

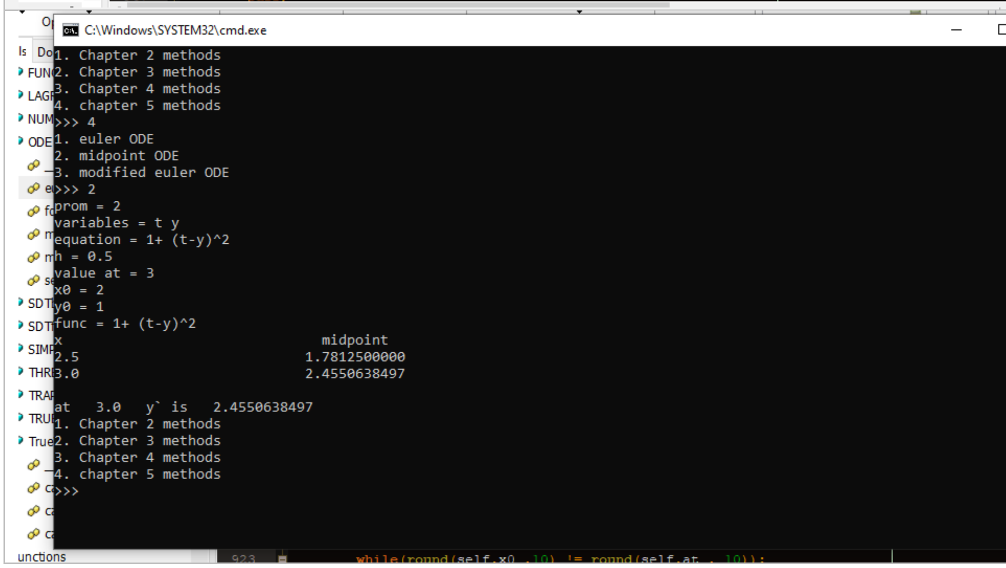
```
1. euler ODE
2. midpoint ODE
3. modified euler ODE
>>> 1
prom = 1
variables = t y
equation = 1 + (y/t)
h = 0.25
value at = 2
x0 = 1
y0 = 2
func = 1 + (y/t)
x                                euler
1.25                           2.7500000000
1.5                            3.5500000000
1.75                           4.3916666667
2.0                            5.2690476190

at   2.0   y` is   5.2690476190
1. Chapter 2 methods
2. Chapter 3 methods
3. Chapter 4 methods
4. chapter 5 methods
>>> 4
1. euler ODE
2. midpoint ODE
3. modified euler ODE
```

Chapter 5 Euler Method Implemented.



Chapter 5 Midpoint Ordinary Differential Equation Implemented.



Chapter 5 Modified Euler Ordinary Differential Equation Implemented.