ASIAN SCHOOL OF MANAGEMENT AND TECHNOLOGY (ASMT)

GONGABU, KATHMANDU

LAB REPORT

NUMERICAL METHOD

SUBMITTED BY:

AASHISH RIJAL
ROLL: 3 (CSIT 3^{nd} SEMESTER)
Reg. No.:5-2-1181-53-2022

SUBMITTED TO:

SURYA BAM
…………….

Table of Content

15.	Write a program to calculate the derivative using central difference formula.
16.	Write a program to calculate the derivative using forward divided difference formula.
17.	Write a program to calculate the derivative using backward divided difference formula.
18.	Write a program to implement trapezoidal rule.
19.	Write a program to implement composite trapezoidal rule.
20.	Write a program to implement Simpson’s 1/3 rule.
21.	Write a program to implement composite Simpson’s 1/3 rule.
22.	Write a program to implement Simpson’s 3/8 rule.
23.	Write a program to implement composite Simpson’s 3/8 rule.
24.	Write a program to implement Gauss Elimination Method.
25.	Write a program to implement Gauss Elimination with pivoting.
26.	Write a program to implement Gauss Jordan Method.
27.	Write a program to implement matrix inversion with Gauss-Jordan Method.
28.	Write a program to implement Do-Little LU Decomposition.
29.	Write a program to implement Cholesky Method.
30.	Write a program to implement Jacobi Iteration Method.
31.	Write a program to implement Gauss Seidal Method.

32.	Write a program to solve ODE by using Taylor’s series Method.
33.	Write a program to solve ODE by using Picard’s Method.
34.	Write a program to solve ODE by using Eulers’s Method.
35.	Write a program to solve ODE by using Heun’s Method.
36.	Write a program to solve ODE by using Taylor’s series Method.
37.	Write a program to solve ODE by using Runge-Kutta Method.
38.	Write a program to solve boundary value problem using Shooting Method.
39.	Write a program to Elliptic PDE.
40.	Write a program to solve Poisson’s Equation.

Lab Assignment #1

1. WAP to implement Bisection Method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-1\Bisection Implementation.exe

 Enter x1 and x2:
-2
-1

 Root is: -1.625000
--------------------------------
Process exited after 27.13 seconds with return value 0
Press any key to continue . . .
```

2. Write a program to implement Secant Method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-1\Secant Method Implemetation.exe

 Enter x1, x2:4
2

 Enter max Iteration6

 step    xo            x1            x2            f(x1)         f(x2)

 1       0.000000      4.000000      0.357143      51.000000          -5.668732

 2       4.000000      0.357143      0.721548      -5.668732          -6.067435

 3       0.357143      0.721548      -4.823938     -6.067435          -107.606979

 4       0.721548      -4.823938     1.052915      -107.606979        -5.938536

 5       -4.823938     1.052915      1.396187      -5.938536          -5.070733

 6       1.052915      1.396187      3.401987      -5.070733          27.568970
Not convergentX2 is root
--------------------------------
```

3. WAP to implement Newton Raphson method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-1\Newton Rapson method.exe

Enter initial guess:
0
Enter tolerable error:
0.0001
Enter maximum iteration:
5

Step            x0              f(x0)           x1              f(x1)
1               0.000000        -2.000000       0.666667        0.000000
2               0.666667        0.214113        0.607493        0.214113
3               0.607493        0.001397        0.607102        0.001397

Root is: 0.607102
```

4. WAP to implement fixed point iteration method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-1\fixed point Impementation.exe
Enter initaial value:0

Enter Interation:4

 step            xo                      f(xo)           x1                      f(x1)

 1               0.000000                2.000000                0.666667                2.000000

 2               0.666667                -0.214113               0.595296                -0.214113

 3               0.595296                0.042096                0.609328                0.042096

 4               0.609328                -0.007950               0.606678                -0.007950

Sorry you meet your Iteration !
```

5. WAP to implement synthetic division.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-1\synthetic division implementation.exe
               SYNTHETIC DIVISION
Enter the highest degree of the equation (max 5): 3

Coefficient x[3] = 1

Coefficient x[2] = -7

Coefficient x[1] = 15

Coefficient x[0] = -9

Enter the value of constant (x) : 3

Coefficients of quotient are:
        1       -4      3
Remainder is: 0
```

6. WAP to implement Horner’s method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-1\Horner's method Implementation.exe

Enter degree of the polynomial X :: 3

Enter coefficient's of the polynomial X ::

Enter Coefficient of [ X^3 ] :: 3

Enter Coefficient of [ X^2 ] :: -4

Enter Coefficient of [ X^1 ] :: 5

Enter Coefficient of [ X^0 ] :: -6

Enter the value of X :: 2

Value of the polynomial is = [ 12.000000 ]
```

Lab Assignment #2

7. WAP to implement Lagrange’s interpolation.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\langarage interpolation implementation.exe
====Langarange Interpolation:
Enter number of data: 6

 x[0] = 0
y[0] = 0

 x[1] = 10
y[1] = 227.04

 x[2] = 15
y[2] = 362.78

 x[3] = 20
y[3] = 517.25

 x[4] = 22.5
y[4] = 602.97

 x[5] = 30
y[5] = 901.67
Enter interpolation points: 16
Interpolated value at 16.000000 is 392.035614
```

8. Write a program to implement Newton's divided difference interpolation.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\Newtons Divided Difference Interpolation.exe
Newton divided difference interpolation:
Enter the number of points:
4
Enter the value of x (point):
2.5
Enter the value of x and fx at i=0:
1
0
Enter the value of x and fx at i=1:
2
0.3010
Enter the value of x and fx at i=2:
3
0.4771
Enter the value of x and fx at i=3:
4
0.6021
Interpolation value = 0.334050
```

9. WAP to implement Newton's forward difference formula.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\newton forward difference interpolation.exe

 Enter value of given xp :
25
Enter the values of x and f(x):
x0= 10
fx0= 0.2736

x1= 20
fx1= 0.3420

x2= 30
fx2= 0.500

x3= 40
fx3= 0.6425

x4= 50
fx4= 0.7110


x0=      10.000000      x1=      20.000000      x2=      30.000000      x3=      40.000000      x4=      50.000000

fx0=     0.273600       fx1=     0.342000       fx2=     0.500000       fx3=     0.642500       fx4=     0.711000

//////The value of h is: 10.000000
/////The value of s is: 1.500000
 The value of f(25.000000) is : 0.417461
--------------------------------
```

10. WAP to implement Newton's backward difference formula.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\newton backward difference interpolation.exe
 =====================Newton's backward difference interpolation ======================

 Enter number of data points:
4

 Enter value of given xp :
5
Enter the values of x and f(x):
x0= 1
fx0= 2

x1= 2
fx1= 3

x2= 3
fx2= 4

x3= 4
fx3= 5


x0=      1.000000       x1=      2.000000       x2=      3.000000       x3=      4.000000
fx0=     2.000000       fx1=     3.000000       fx2=     4.000000       fx3=     5.000000
//////The value of h is: 1.000000
/////The value of s is: 1.000000
 The value of f(5.000000) is : 11.000000
```

11. WAP to implement least square approximation.
    a. Linear least square method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\Least square approximation(regression).exe
***Least square method***
Input the number of data points:
n=4
Enter the data sets one after another:
x[0]=1

y[0]=6

x[1]=2

y[1]=11

x[2]=3

y[2]=18

x[3]=4

y[3]=27
The equation of line:

 y=7.000000x+-2.000000
```

Polynomial regression.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\Second order polynomial (regression).exe
***Least square method for second-order polynomial***
Input the number of data points: 5
Enter the data sets one after another:
x[0]=1
y[0]=3
x[1]=2
y[1]=5
x[2]=3
y[2]=7
x[3]=4
y[3]=10
x[4]=5
y[4]=12

The equation of the second-order polynomial:

y = -0.020620 * x^2 + -0.943896 * x + -3.010144
```

b. Exponential regression.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\Exponential Regression.exe
Enter the number of data points: 6
Enter the value of x and y respectively:
x[0]= 0.4
y[0]= 0.8
x[1]= 1.2
y[1]= 1.6
x[2]= 2
y[2]= 2.3
x[3]= 3
y[3]= 2.6
x[4]= 4
y[4]= 2.8
x[5]= 6
y[5]= 3.1
y = 1.131291 * e^(0.208202 * x)
```

12. WAP to implement maxima and minima of tabulated function.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-2\maxima and minima of table value.exe
Enter the number of data points: 4
Enter the data points (x, y):
x[0] = 0
y[0] = -5
x[1] = 1
y[1] = -7
x[2] = 2
y[2] = -3
x[3] = 3
y[3] = 13
Maxima and Minima:
Minima at (x[1] = 1.000000, y[1] = -7.000000)
```

Lab Assignment #3

1. Write a program to calculate the derivative using forward difference

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\1.forward difference appr  taylor.exe

 Forward Difference Approximation using Taylor series:
Enter initial point x: 1
Enter difference point h: 0.2
The forward difference approximation of the derivative is: 2.200000

--------------------------------
Process exited after 109 seconds with return value 0
Press any key to continue . . .
```

2. Write a program to calculate the derivative using backward difference

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\2.backward difference appr taylor.exe

 Backward Difference Approximation using Taylor series:
Enter initial point x: 1
Enter difference point h: 0.2
The backward difference approximation of the derivative is: 1.800000

--------------------------------
```

3. Write a program to calculate the derivative using central difference formula.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\3.Central difference appr taylor.exe

 Central Difference Approximation using Taylor series:
Enter initial point x: 1
Enter difference point h: 0.2
The central difference approximation of the derivative is: 2.000000
```

4. Write a program to calculate the derivative using forward divided difference

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\5. derivative forward.exe
Derivative using newton forward difference:
========================

Enter the number of data points: 6
Enter the data points in the format (x, y):
x[0] = 1
y[0] = 0
x[1] = 1.2
y[1] = 0.128
x[2] = 1.4
y[2] = 0.554
x[3] = 1.6
y[3] = 1.296
x[4] = 1.8
y[4] = 2.432
x[5] = 2
y[5] = 4
Enter the value at which you want to calculate the derivative: 1.1
The derivative at x = 1.100000 is: 0.640000
```

5. program to calculate the derivative using backward divided difference

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\6.Derivative Backward.exe
Derivative using newton backward difference:
========================

Enter the number of data points: 5
Enter the data points in the format (x, y):
x[0] = 5
y[0] = 10
x[1] = 6
y[1] = 15
x[2] = 7
y[2] = 20
x[3] = 8
y[3] = 23
x[4] = 9
y[4] = 32
Enter the value at which you want to calculate the derivative: 9
The derivative at x = 9.000000 is: 3.000000
```

6. Write a program to implement trapezoidal rule.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\8. Trapezoidal function.exe

        ====Trapezoidal===
Please enter the limits of integration
2
8
The value of integration is
It= 1566.000000
--------------------------------
```

7. Write a program to implement composite trapezoidal rule.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\9.composite trapezoidal(fun).exe


                  Composite trapezoidal(fun)
                 ====================================

 Enter the initial value of x(a):-1

Enter the final value of x(b):1

Enter the segment number(n):2

The integration value of function =4.086162
```

8. Write a program to implement Simpson’s **1/3** rule **.**

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\10. simpson 1 by3 (fun).exe
                 === Simpson 1/3(fun)==
Enter initial value of X:
a=0

Enter  Final  value of X:
b=1

Enter number of segments (Even number):
N=3

Integral from 0.000000 to 1.000000
When h=0.333333 is 0.000000
```

9. Write a program to implement composite Simpson’s 1/3 rule

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\11. simpson 1by3 composite(fun).exe

                ==== Composite Simpson 1/3(fun)=====
Enter initial value of X: 0
Enter final value of X: 1
Enter number of segments (Even number): 4

Integral from 0.000000 to 1.000000 when h = 0.250000 is 0.213060
```

10. Write a program to implement Simpson’s 3/8 rule**.**

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\13.Simpson 3 by 8(fun).exe

                 === Simpson 3/8(fun)====

Enter the initial value of x:0

Enter the initial value of x:1

Integration value:0.462073
```

11. Write a program to implement composite Simpson’s 3/8 rule.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\Unit-3\14. Simpson 3by8 composite(fun).exe

                === Composite Simpson 3/8 function=====

Enter the initial value of x:
4

Enter the final value of x: 5.2

Enter the number of segments (a multiple of 3): 3

Integration value: -1.122533
```

Lab Assignment #4

1. Write a Program to implement Gauss Elimination Method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\1.gaussElimination.exe
         ===GAuss Elimination ====
Enter number of unknowns: 3
a[1][1] = 20
a[1][2] = 15
a[1][3] = 10
a[1][4] = 45
a[2][1] = -3
a[2][2] = 2.24
a[2][3] = 7
a[2][4] = 1.75
a[3][1] = 5
a[3][2] = 1
a[3][3] = 5
a[3][4] = 9

Solution:
x[1] = 1.183
x[2] = 1.167
x[3] = 0.384
```

2. Write a Program to implement Gauss Elimination with pivoting.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\2.Gauss Elmination with pivoting.exe

                                ========Gauss Elimination With Pivoting

 Enter matrix a
 a11, a12, a13
 a21, a22, a23
 a31, a32, a33

20 15 10

-3 -2.2 7

5 1 5

 Enter matrix b
 b1, b2, b3
45 1.7 9

Solution:
x[1] = 0.473409
x[2] = 1.712759
x[3] = 0.984043
```

3. Write a Program to implement Gauss Jordan Method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\3.Gauss jordan Mthod.exe
         =====GAUSS JORDAN====
Enter the number of unknowns: 3
Enter coefficients of the Augmented Matrix:
a[1][1] = 2
a[1][2] = -
a[1][3] = 4
a[1][4] = 15
a[2][1] = 2
a[2][2] = 3
a[2][3] = -2
a[2][4] = 1
a[3][1] = 3
a[3][2] = 2
a[3][3] = -4
a[3][4] = -4

Solution:
x[1] = 1.778
x[2] = 1.056
x[3] = 2.861
```

4. Write a Program to implement matrix inversion with Gauss-Jordan method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\4.Inverse matrix.exe
                         ====Gauss Jordan Method for Matrix Inversion
Enter order of matrix: 3
Enter coefficients of Matrix:
a[1][1] = 1
a[1][2] = -1
a[1][3] = 1
a[2][1] = 2
a[2][2] = 3
a[2][3] = 0
a[3][1] = 0
a[3][2] = -2
a[3][3] = 1

Inverse Matrix is:
3.000   -1.000  -3.000
-2.000  1.000   2.000
-4.000  2.000   5.000
```

5. Write a Program to implement Do-Little LU Decomposition.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\5.LU decomposition.exe
                         ===== Doolittle LU Decomposition========
Enter the elements of the 3x3 matrix:
25 5 1
64 8 1
144 12 1

Lower Triangular Matrix (L):
1.00    0.00    -1.#R
2.56    1.00    0.00
5.76    4.00    1.00

Upper Triangular Matrix (U):
25.00   5.00    1.00
0.00    -4.00   -1.00
0.00    0.00    0.00
```

6. Write a Program to implement Cholesky Method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\6.cholesky decomposition.exe
                                 =======Cholesky Decomposition method=====
Enter the size of the matrix: 3
Enter the elements of the matrix:
1 4 7
4 80 44
7 44 89

Lower Triangular Matrix (L):
Upper metrix(U) is the transpose metrix of lower matrix
1.000 0.000 0.000
4.000 8.000 0.000
7.000 2.000 6.000
```

7. Write a Program to implement Jacobi iteration method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\7.Jacobbi Iteration.exe
         ===JACOBIAN ITERATION==
Enter the coefficients of the matrix A and the vector B:
Row 1 (separate each coefficient by a space): 6 -2 1 11
Row 2 (separate each coefficient by a space): -2 7 2 5
Row 3 (separate each coefficient by a space): 1 2 -5 -1
Enter initial guess for solution X:
0 0 0

Solution:
X1 = 1.999973
X2 = 0.999958
X3 = 1.000047

Number of iterations: 10
```

8. Write a Program to implement Gauss Seidal Method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-4\8.Gauss-seidal.exe
                         ===== Gauss_seidal========
Enter the size of the matrix: 3
Enter the elements of the matrix:
6
-2
1
-2
7
2
1
2
-5
Enter the elements of the right-hand side vector:
11 5 -1
Enter the maximum number of iterations: 7
Enter the value of epsilon: 0.001
Enter the initial guess for the solution vector:
0 0 0

Solution Vector (x):
2.000014
0.999990
0.999999
```

Lab Assignment #5

1. Write a Program to solve ODE by Using Taylors series method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\1.ODE Taylor.exe

                ======ODE using Taylor Series Method======
Enter initial value of t (t0): 0.5
Enter initial value of y (y0): 1
Enter step size (h): 0.25
Enter final value of t (T): 4
The approximate value of y(4.000000) is 110.541047
```

2. Write a Program to solve ODE by Using Picard's method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\2.Picards.exe
Picard Iteration Method for ODE solving
Enter initial value of t (t0): 0
Enter initial value of y (y0): 1
Enter step size (h): 1
Enter final value of t (T): 0.2
The approximate value of y(0.200000) is 2.000000
```

3. Write a Program to solve ODE by Using Eulers's method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\3. euler.exe
         ==EULER==
Input initial value of x and y0
1
Input X-value at which Y is required :
0.5
Input  step size:
0.1
x=0.100000 and y=2.000000
x=0.200000 and y=4.020000
x=0.300000 and y=8.080000
x=0.400000 and y=16.220001
x=0.500000 and y=32.520004

 Value of y at x =0.500000 is 32.520004
```

4. Write a Program to solve ODE by Using Heun's method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\4.heuns.exe
         ==HEUN'S==

Input initialvalue of x and y;
0
1
Input X-value at which y is required:0.4

Input stop size:0.1

x=0.100000 n y=1.110000
x=0.200000 n y=1.252550
x=0.300000 n y=1.431068
x=0.400000 n y=1.649330
Value of y at x=0.400000 is 1.649330
```

5. Write a Program to solve ODE by Using Runge-Kutta method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\5.Rk.exe
        ===RK===
Input initial value of x and y0
1
Input X-value at which Y is required :
0.4
Input  step size:
0.4
x=0.400000 and y=1.583467

 Value of y at xp =0.400000 is 1.583467
--------------------------------
```

6. Write a program to solve boundary value problem using shooting method.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\6.shooting.exe
         === ODE using Shouting method ===
Enter the initial value of x: 1
Enter the initial value of y (boundary condition y(x0)): 2
Enter the final value of x: 2
Enter the desired boundary value y(xn): 9
The value of z at x=0 is: 12.029800
The value of y at x=1 is: 9.000000
```

7. Write a program to Elliptic PDE.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\7.elliptic.exe
Enter the number of interior grid points in the x direction: 3
Enter the number of interior grid points in the y direction: 3
Enter the grid spacing in the x direction (cm): 5
Enter the grid spacing in the y direction (cm): 5
Enter the temperature at the top boundary (░C): 0
Enter the temperature at the bottom boundary (░C): 0
Enter the temperature at the left boundary (░C): 100
Enter the temperature at the right boundary (░C): 100

Steady-state temperature distribution after 1 iterations:
100.00░C        0.00░C  100.00░C
100.00░C        50.00░C 100.00░C
100.00░C        0.00░C  100.00░C
```

8. Write a program to solve poisson's equation.

```
C:\Users\Dell\OneDrive\Desktop\Aashis\Numerical_Method\codes\unit-5\7.elliptic.exe
Enter xl: 0
Enter xh: 1
Enter yl: 0
Enter boundary conditions:
a[1,1] = 0
a[1,2] = 20
a[1,3] = 10
a[2,1] = 30
a[2,3] = 5
a[3,1] = 15
a[3,2] = 5
a[3,3] = 5
Solution:
0.00    20.00   10.00
30.00   15.25   5.00
15.00   5.00    5.00

--------------------------------
```