Question 1: *// Write a Program to implement Numerical Differentiation using Newton's Forward difference Formula*

*// 20220PHY014*

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{

    float x[20], y[20][20], find\_derivative, h, sum\_of\_terms = 0.0, term, first\_derivative; *// 'h' is finite difference*

    int total\_given\_values, index, check\_validation = 0, sign = 1;                          *// 'sign' for alternate +- sign used in formula*

    printf("Enter total given number of data points: ");

    scanf("*%d*", &total\_given\_values);

    for (int i = 0; i < total\_given\_values; i++)

    {

        printf("Enter the value of x[*%d*] = ", i);

        scanf("*%f*", &x[i]);

        printf("Enter the value of y[*%d*] = ", i);

        scanf("*%f*", &y[i][0]);

        printf("\n");

    }

    h = x[1] - x[0];

    printf("Enter the value at which derivative of the function to be found: ");

    scanf("*%f*", &find\_derivative);

*// Checking for validation of given point*

    for (int i = 0; i < total\_given\_values; i++) *// it gives x[i] which is closest to find\_derivative*

    {

        if (fabs(find\_derivative - x[i]) < 0.0001)

        {

            index = i; *// index of calculation point*

            check\_validation = 1;

            break;

        }

    }

*// if given point is not a valid point i.e. its value is varies much from x[i]*

    if (check\_validation == 0)

    {

        printf("Next time try with different data point!!!\n");

        exit(0);

    }

*// Forming Forward difference table*

    for (int i = 1; i < total\_given\_values; i++)

        for (int j = 0; j < total\_given\_values - i; j++)

            y[j][i] = y[j + 1][i - 1] - y[j][i - 1];

*// Applying formula to find differentiation*

    for (int i = 1; i < total\_given\_values; i++)

    {

        term = pow(y[index][i], i) / i;

        sum\_of\_terms += sign \* term;

        sign = -sign;

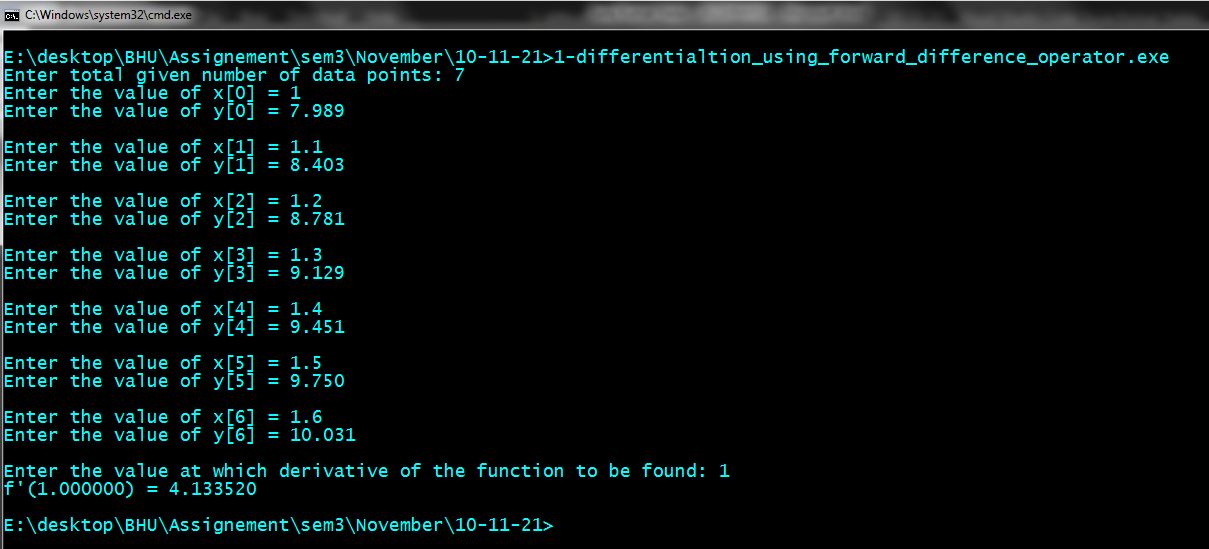
    }

    first\_derivative = sum\_of\_terms / h; *// finally dividing by 'h' to get the result*

    printf("f'(*%f*) = *%f*\n", find\_derivative, first\_derivative);

    return 0;

}

Output: 

Question 2: *// Write a Program to implement Numerical Differentiation using Newton's Backward difference Formula*

*// 20220PHY014*

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{

    float x[20], y[20][20], find\_derivative, h, sum\_of\_terms = 0.0, first\_derivative, term; *// 'h' is finite difference*

    int total\_given\_values, index, check\_validation = 0;

    printf("Enter total given number of data points: ");

    scanf("*%d*", &total\_given\_values);

    for (int i = 0; i < total\_given\_values; i++)

    {

        printf("Enter the value of x[*%d*] = ", i);

        scanf("*%f*", &x[i]);

        printf("Enter the value of y[*%d*] = ", i);

        scanf("*%f*", &y[i][0]);

        printf("\n");

    }

    h = x[1] - x[0];

    printf("Enter the value at which derivative of the function to be found: ");

    scanf("*%f*", &find\_derivative);

*// Checking for validation of given point*

    for (int i = 0; i < total\_given\_values; i++)

    {

        if (fabs(find\_derivative - x[i]) < 0.0001)

        {

            index = i; *// index of calculation point*

            check\_validation = 1;

            break;

        }

    }

*// if given point is not a valid point i.e. its value is varies much from x[i]*

    if (check\_validation == 0)

    {

        printf("Next time try with different data point!!!\n");

        exit(0);

    }

*// Forming Backward difference table*

    for (int i = 1; i < total\_given\_values; i++)

        for (int j = total\_given\_values - 1; j > i - 1; j--)

            y[j][i] = y[j][i - 1] - y[j - 1][i - 1];

*// Applying formula to find differentiation*

    for (int i = 1; i <= index; i++)

    {

        term = pow(y[index][i], i) / i;

        sum\_of\_terms += term;

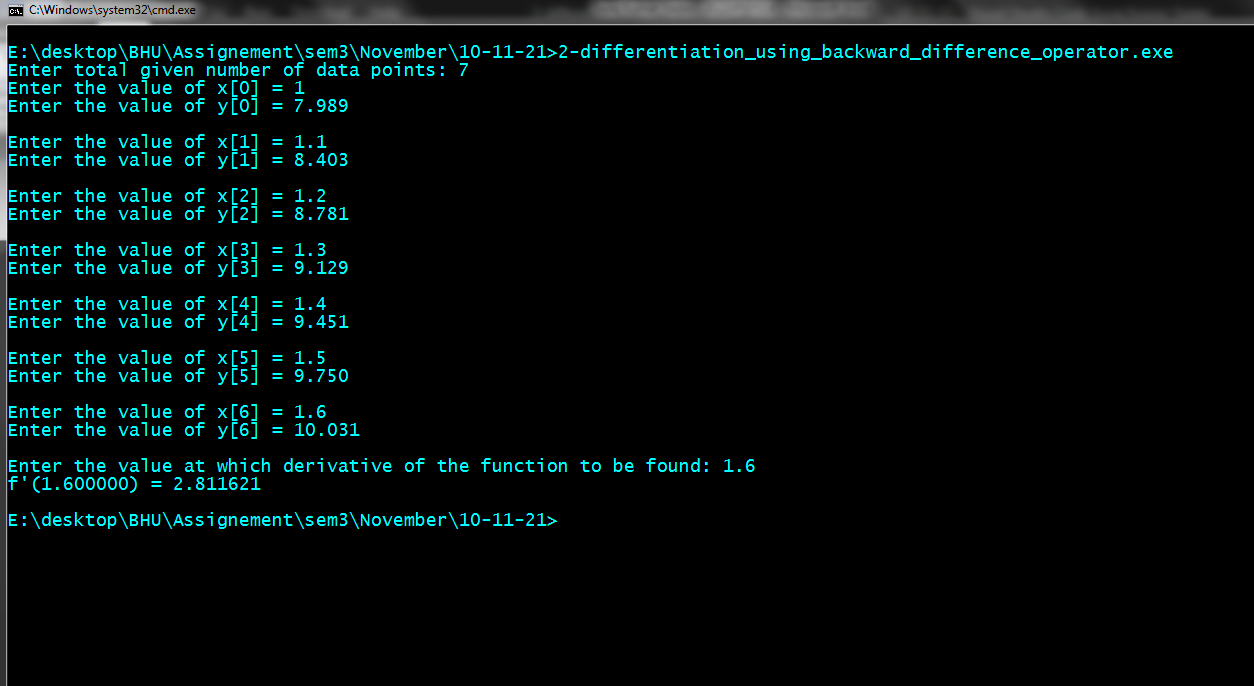
    }

    first\_derivative = sum\_of\_terms / h; *// finally dividing by 'h' to get the result*

    printf("f'(*%f*) = *%f*\n", find\_derivative, first\_derivative);

    return 0;

}

Output: 

Question 3: *// Write a Program to implement Trapezoidal rule for finding the integral of a given function*

*// 20220PHY014*

#include <stdio.h>

#include <math.h>

float f(float *x*)

{

    return 1 / (1 + pow(*x*, 2));

}

int main()

{

    float lower\_limit, upper\_limit, h, sum, f(float);

    int equal\_width;

    printf("Enter the lower limit: ");

    scanf("*%f*", &lower\_limit);

    printf("Enter the upper limit: ");

    scanf("*%f*", &upper\_limit);

    printf("Enter the value of equal width: ");

    scanf("*%d*", &equal\_width);

    h = (upper\_limit - lower\_limit) / equal\_width;

    sum = (f(lower\_limit) + f(upper\_limit)) / 2;

    for (int i = 1; i < equal\_width; i++)

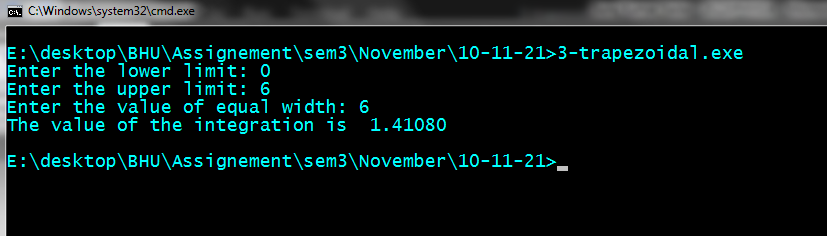
        sum += f(lower\_limit + i \* h);

    sum \*= h;

    printf("The value of the integration is *%8.5f*\n", sum);

    return 0;

}

Output: 

Question 4: *// Write a Program to implement Simpson's 1/3 rule for finding the integral of a given function*

*// 20220PHY014*

#include <stdio.h>

#include <math.h>

float f(float *x*)

{

    return 1 / (1 + pow(*x*, 2));

}

int main()

{

    float lower\_limit, upper\_limit, sum, h, f(float);

    int equal\_width;

    printf("Enter the lower limit: ");

    scanf("*%f*", &lower\_limit);

    printf("Enter the upper limit: ");

    scanf("*%f*", &upper\_limit);

    printf("Enter the value of equal width: ");

    scanf("*%d*", &equal\_width);

    h = (upper\_limit - lower\_limit) / equal\_width;

    sum = f(lower\_limit) + f(upper\_limit);

    for (int i = 1; i < equal\_width; i++)

    {

        if (i % 2 == 0)

            sum += 2 \* f(lower\_limit + i \* h);

        else

            sum += 4 \* f(lower\_limit + i \* h);

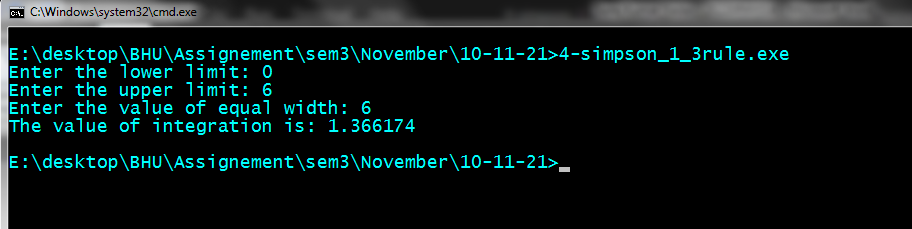
    }

    sum \*= (h / 3);

    printf("The value of integration is: *%f*\n", sum);

    return 0;

}

Output: 

Question 5: *// Write a Program to implement Simpson's 3/8 rule for finding the integral of a given function*

*// 20220PHY014*

#include <stdio.h>

#include <math.h>

float f(float *x*)

{

    return 1 / (1 + pow(*x*, 2));

}

int main()

{

    float lower\_limit, upper\_limit, sum, h, f(float);

    int equal\_width;

    printf("Enter the lower limit: ");

    scanf("*%f*", &lower\_limit);

    printf("Enter the upper limit: ");

    scanf("*%f*", &upper\_limit);

    printf("Enter the value of equal width: ");

    scanf("*%d*", &equal\_width);

    h = (upper\_limit - lower\_limit) / equal\_width;

    sum = f(lower\_limit) + f(upper\_limit);

    for (int i = 1; i < equal\_width; i++)

    {

        if (i % 3 == 0)

            sum += 2 \* f(lower\_limit + i \* h);

        else

            sum += 3 \* f(lower\_limit + i \* h);

    }

    sum \*= ((h \* 3) / 8);

    printf("The value of integration is: *%f*\n", sum);

    return 0;

}

Output: 