**PROGRAM: 13**

**Name: Kiran Kanyal**

**Roll No: 29**

**Section: A1**

**Aim: Write a C program to Integrate numerically using Trapezoidal rule.**

**ALGORITHM:**

**START**

1. Prompt the user to input the number of intervals, n.
2. Prompt the user to input the lower limit a and the upper limit b.
3. Compute the step size h as:

1. Initialize an array y to store the function values at each interval point:

for i=0 to n:

compute y[i] = f(a+i\*h)

1. Initialize a variable res to 0 to store the result of the summation.
2. Compute the summation using the Trapezoidal Rule.
3. Compute the final result as: res = (h/2)\*res
4. Print the computed integral value res.

**STOP**

**PROGRAM:**

#include<stdio.h>

#include<math.h>

float f(float x){

return 1/(1+pow(x,2));

}

#include<stdio.h>

int main(){

int n;

printf("Enter the number of intervals: ");

scanf("%d", &n);

float a,b;

printf("Enter the limit: ");

scanf("%f%f", &a, &b);

float h = (b-a)/n;

float y[n+1];

for (int i=0; i<=n; i++){

y[i] = f(a+i\*h);

}

float res = 0;

for (int i=0; i<=n; i++){

if (i==0 || i==n){

res += y[i];

}

else {

res += 2\*y[i];

}

}

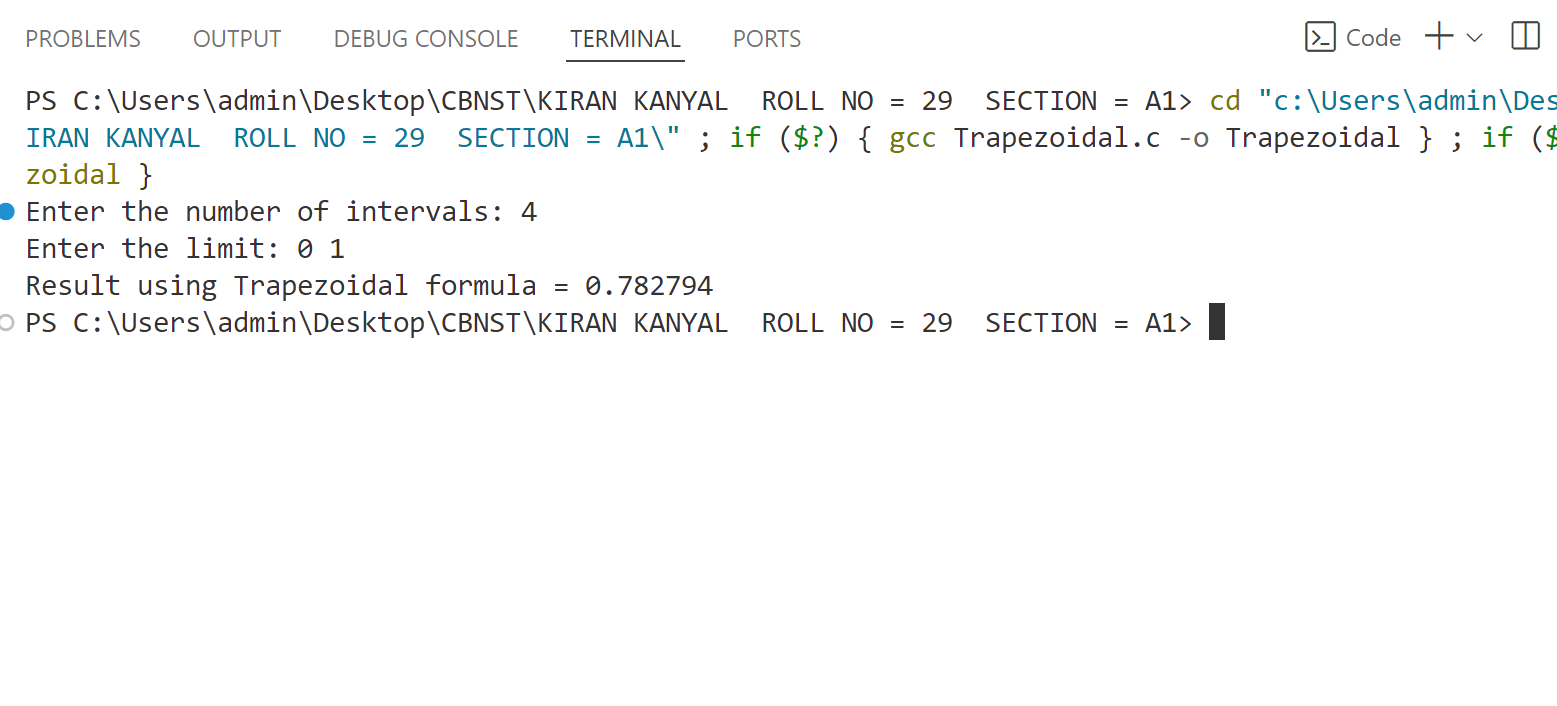
res = (h/2)\*res;

printf("Answer = %f", res);

return 0;

}

**OUTPUT**



**PROGRAM: 14**

**Name: Kiran Kanyal**

**Roll No: 29**

**Section: A1**

**Aim: Write a C program to Integrate numerically using Simpson’s 1/3 rules.**

**ALGORITHM:**

**START**

1. Prompt the user to input the number of intervals, n. Ensure that n is even, as Simpson’s 1/3 Rule requires an even number of intervals.
2. Prompt the user to input the lower limit a and the upper limit b.
3. Compute the step size h as:

1. Initialize an array y to store the function values at each interval point:

for i=0 to n:

compute y[i] = f(a+i\*h)

1. Initialize a variable res to 0 to store the result of the summation.
2. Compute the summation using the Simpson’s 1/3 Rule.
3. Compute the final result as: res = (h/3)\*res
4. Print the computed integral value res.

**STOP**

**PROGRAM:**

#include<stdio.h>

#include<math.h>

float f(float x){

return 1/(1+pow(x,2));

}

int main(){

int n;

printf("Enter the no. of intervals: ");

do{

scanf("%d",&n);

if (n%2!=0) printf("Number of intervals should be even.\n");

}while (n%2!=0);

float a,b;

printf("Enter the limit: ");

scanf("%f%f", &a, &b);

float h = (b-a)/n;

float y[n+1];

for (int i=0; i<=n; i++){

y[i] = f(a+i\*h);

}

float res = 0;

for (int i=0; i<=n; i++){

if (i==0 || i==n){

res += y[i];

}

else if (i%2 == 0){

res += 2\*y[i];

}

else{

res += 4\*y[i];

}

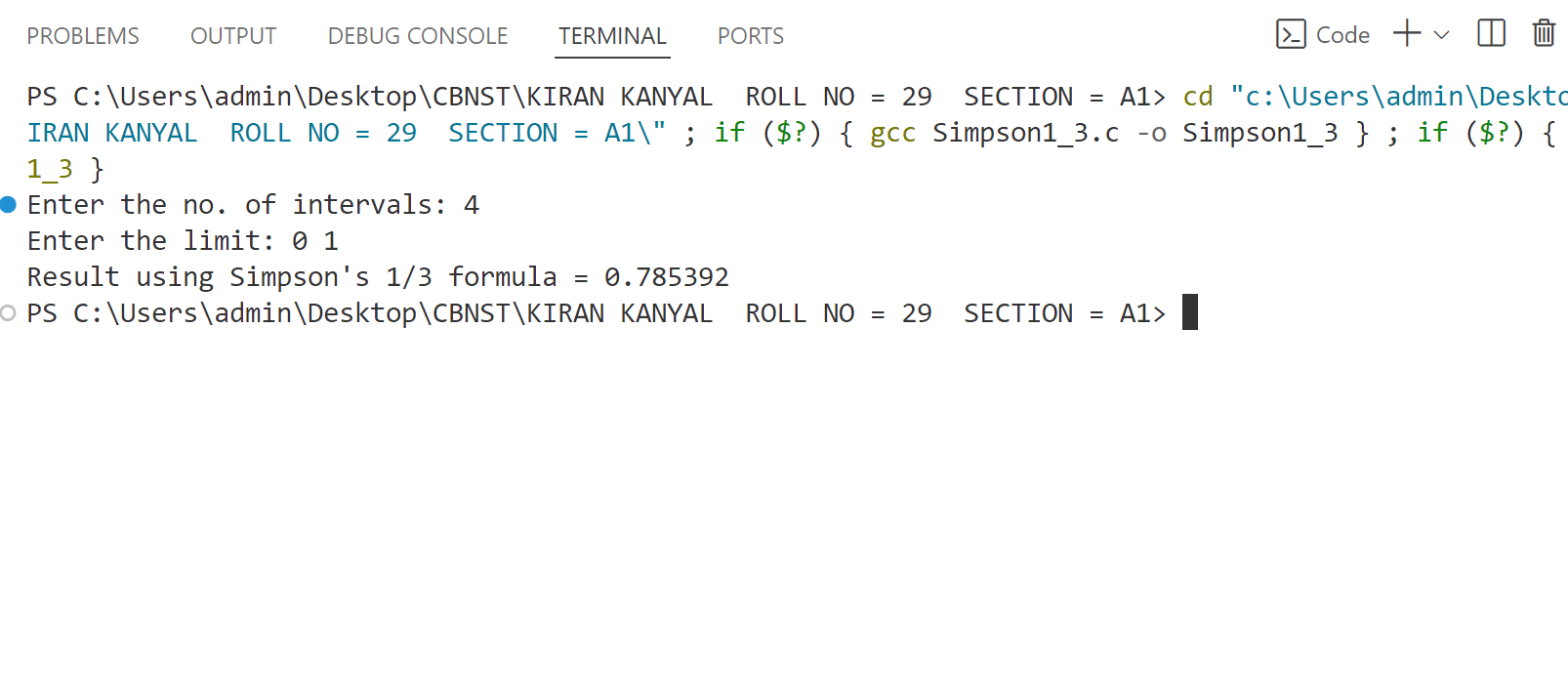
}

res = res\*(h/3);

printf("Answer = %f",res);

}

**OUTPUT:**



**PROGRAM: 15**

**Name: Kiran Kanyal**

**Roll No: 29**

**Section: A1**

**Aim: Write a C program to Integrate numerically using Simpson’s 3/8 rules.**

**ALGORITHM:**

**START**

1. Prompt the user to input the number of intervals, n. Ensure that n is a multiple of 3, as Simpson’s 3/8 Rule requires it.
2. Prompt the user to input the lower limit a and the upper limit b.
3. Compute the step size h as:

1. Initialize an array y to store the function values at each interval point:

for i=0 to n:

compute y[i] = f(a+i\*h)

1. Initialize a variable res to 0 to store the result of the summation.
2. Compute the summation using Simpson’s 3/8 Rule.
3. Compute the final result as: res = (3h/8)\*res
4. Print the computed integral value res.

**STOP**

**PROGRAM:**

#include<stdio.h>

#include<math.h>

float f(float x){

return 1/(1 + pow(x,2));

}

int main(){

int n;

do {

printf("Enter the number of intervals:");

scanf("%d",&n);

if (n%3!=0){

printf("Number of intervals should be multiple of 3.");

}

}while (n%3!=0);

float a,b;

printf("Enter the limits: ");

scanf("%f%f", &a, &b);

float h = (b-a)/n;

float y[n+1];

for (int i=0; i<=n; i++){

y[i] = f(a+i\*h);

}

float res = 0;

for (int i=0; i<=n; i++){

if (i==0 || i==n){

res += y[i];

}

else if (i%3==0){

res += 2\*y[i];

}

else{

res+= 3\*y[i];

}

}

res = res\*(3\*h/8);

printf("Answer = %f",res);

}

**OUTPUT:**

