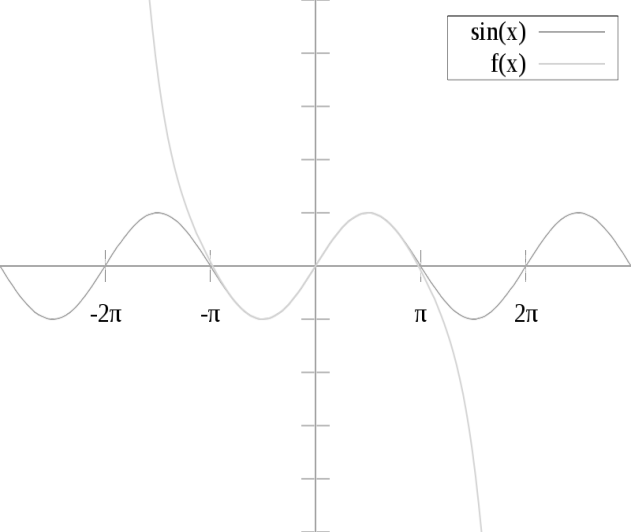
**Date-**

**Assignment No. :**

**Problem Statement:**

Program in C to find the value of sine function for a particular angle given in degree corrected upto 4 decimal places.

**Theory:**

In [mathematics](https://en.wikipedia.org/wiki/Mathematics), a Taylor series is a representation of a [function](https://en.wikipedia.org/wiki/Function_(mathematics)) as an [infinite sum](https://en.wikipedia.org/wiki/Series_(mathematics)) of terms that are calculated from the values of the function's [derivatives](https://en.wikipedia.org/wiki/Derivative) at a single point. A function can be approximated by using a finite number of terms of its Taylor series. [Taylor's theorem](https://en.wikipedia.org/wiki/Taylor%27s_theorem) gives quantitative estimates on the error introduced by the use of such an approximation. The polynomial formed by taking some initial terms of the Taylor series is called a Taylor polynomial. The Taylor series of a function is the [limit](https://en.wikipedia.org/wiki/Limit_of_a_sequence) of that function's Taylor polynomials as the degree increases, provided that the limit exists. A function may not be equal to its Taylor series, even if its Taylor series [converges](https://en.wikipedia.org/wiki/Convergence_(mathematics)) at every point. A function that is equal to its Taylor series in an [open interval](https://en.wikipedia.org/wiki/Open_interval) (or a [disc](https://en.wikipedia.org/wiki/Disk_(mathematics)) in the [complex plane](https://en.wikipedia.org/wiki/Complex_plane)) is known as an [analytic function](https://en.wikipedia.org/wiki/Analytic_function) in that interval.

Pictured on the right is an accurate approximation of sin x around the point x = 0. The pink curve is a polynomial of degree seven:

{\displaystyle \sin \left(x\right)\approx x-{\frac {x^{3}}{3!}}+{\frac {x^{5}}{5!}}-{\frac {x^{7}}{7!}}.\!}Sinx≈ x - x3/3! + x5/5! +…..

The error in this approximation is no more than |x|9/9! . In particular, for −1 < x < 1, the error is less than 0.000003.

In contrast, also shown is a picture of the natural logarithm function [log](https://en.wikipedia.org/wiki/Natural_logarithm)(1 + x) and some of its [Taylor polynomials](https://en.wikipedia.org/wiki/Taylor_polynomial) around a = 0. These approximations converge to the function only in the region −1 < x ≤ 1; outside of this region the higher-degree Taylor polynomials are worse approximations for the function. This is similar to [Runge's phenomenon](https://en.wikipedia.org/wiki/Runge%27s_phenomenon" \o "Runge's phenomenon).

The error incurred in approximating a function by its nth-degree Taylor polynomial is called the remainder or [residual](https://en.wikipedia.org/wiki/Residual_(numerical_analysis)) and is denoted by the function Rn(x). [Taylor's theorem](https://en.wikipedia.org/wiki/Taylor%27s_theorem) can be used to obtain a bound on the size of the remainder.

**Algorithm:**

**Input specification:** The magnitude of the angle for which sin series is to be calculated say, X.

**Output specification:** 1. Value by sin series and value by library function.

2. Error calculation by taking the mod value of the difference between library function value and sin series value.

**Steps:**

1. [Starting of Do-While loop]

Set t= (3.141593\*x)/180

1. Set sine=t
2. Set sign=-1
3. Set index=3
4. Repeat through step 6 to step 15 While(numer /denom<0.00001)
5. Set numer=pow(t,index)
6. Set denom=1
7. Repeat step 9 to 10 For j=1 to index
8. Set denom=denom\*j
9. Set i=i+1

[End of For loop]

1. Set term=(sign\*numer)/denom
2. Set sine=sine+term
3. Set index=index+2
4. Set sign=sign\*-1
5. Set c=sin(t) //sin(t) is a inbuilt library function which is used to

//calculate the value of sin function corresponding to degree t

1. Set error=sine-c
2. Print "Value of sine"x
3. Print "As per series="sine
4. Print "As per library function "c
5. Print "Error ="error
6. Print "Do you want to continue:(Y/N)?"
7. Input ch
8. If(ch=='n' OR ch=='N') Then
9. Stop
10. Repeat through step 1 to step 24 While (ch=='y' OR ch=='Y')

[End of Do-While loop]

1. End

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

char ch;

do

{

int i, j, denom, index, sign;

float term, sum, x, sine, numer, error, c,t;

printf("Enter the value of x:");

scanf("%f",&x);

t= (3.141593\*x)/180; //converting degree to radian

sine=t;

sign=-1;

index=3;

while(1) //Sin series calculation

{

numer=pow(t,index);

denom=1;

for(j=1;j<=index;j++)

denom=denom\*j;

term=(sign\*numer)/denom;

if(numer /denom<0.00001) //Precision checker

break;

sine=sine+term;

index=index+2;

sign=sign\*-1;

}

c=sin(t);

error=fabs(sine-c); //Absolute error calculation

printf("\nValue of sine %f",x);

printf("\nAs per series=%f",sine);

printf("\nAs per library function %f", c);

printf("\nError =%f",error);

printf("\nDo you want to continue:(Y/N)?"); //Continuity

fflush(stdin);

scanf(" %c",& ch);

if(ch=='n'|| ch=='N')

return 0;

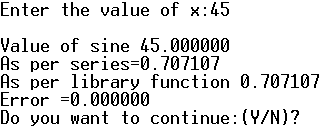
} while (ch=='y'|| ch=='Y');

return 0;

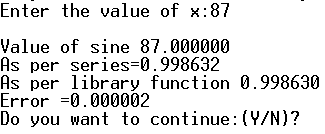
}

**Input & Output:**

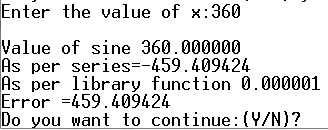
Set 1:



Set 2:



Set 3:



**Discussion:**

1. We are going to assume that the series representation will converge to *f(x)http://tutorial.math.lamar.edu/Classes/DE/FourierSineSeries_files/empty.gif*on *–L ≤ x ≤ L*http://tutorial.math.lamar.edu/Classes/DE/FourierSineSeries_files/empty.gif.  We will be looking into whether or not it will actually converge in a later [section](http://tutorial.math.lamar.edu/Classes/DE/ConvergenceFourierSeries.aspx).
2. The series representation will not involve powers of sine (again contrasting this with Taylor Series) but instead will involve sine’s with different arguments.
3. In set 3 when the input angle crosses 90 the error becomes higher. It is because the higher terms for which the sin function needs to be calculated for that angle are ignored. That’s why when x=180 or 360 then error is very high.