COMPUTER PROGRAMING

EX NO - 3

Name: Kakavakam Jaswanth Sai

Roll no: CH.EN.U4CSE20130

Section: CSE-B

Subject code: 19CSE102 LAB

EX NO -3: SIMPLE PROGRAMS USING FUNCTION

A. SUM OF PROPER DIVISORS.

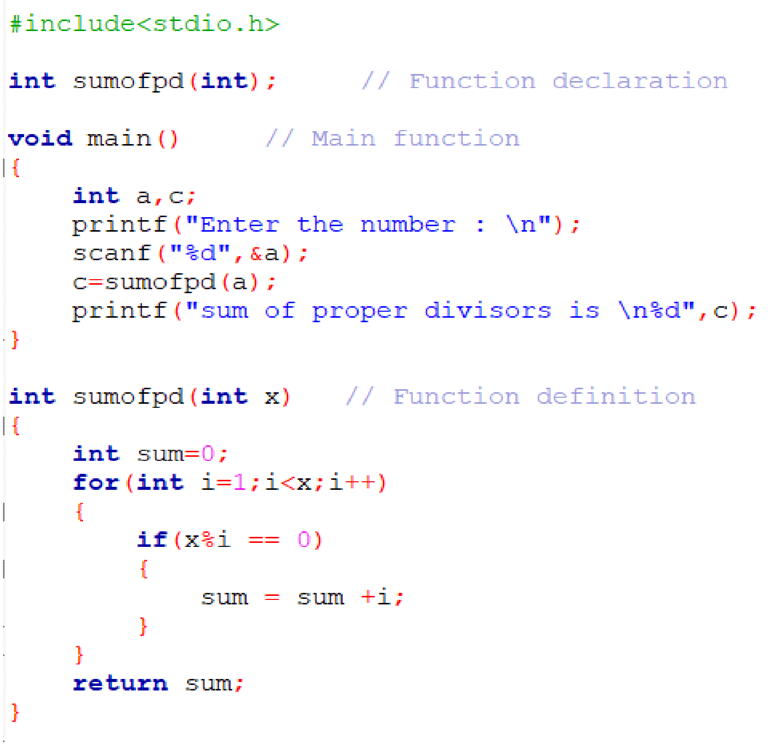
Aim:

To Write a program in C to find sum of the proper divisors using Functions.

Algorithm:

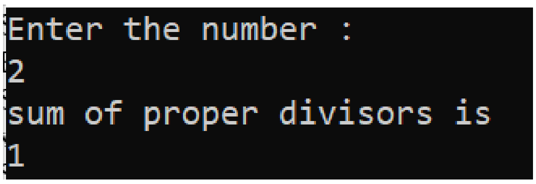
```
MAIN:
    Step 1: Start
    Step 2: Declare two variables a, c .
    Step 3: input the value of a and call the function Sumofpd(a)
    Step 4: Store the value of Sumofpd(a) to c.
    Step 5: Print the value of c.
    Step 6: Stop
Sumofpd(x):
    Step 1: Start
    Step 2: Declare int variable sum and initialize to zero.
    Step 3: For (int i=1; i<x; i++)
                If (x% == 0)
                    Sum = sum;
    Step 4: Return sum.
```

Program:

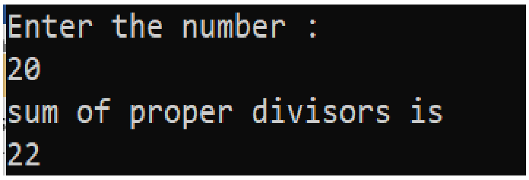


```c
#include<stdio.h>

int sumofpd(int);       // Function declaration

void main()       // Main function
{
    int a,c;
    printf("Enter the number : \n");
    scanf("%d",&a);
    c=sumofpd(a);
    printf("sum of proper divisors is \n%d",c);
}

int sumofpd(int x)    // Function definition
{
    int sum=0;
    for(int i=1;i<x;i++)
    {
        if(x%i == 0)
        {
            sum = sum +i;
        }
    }
    return sum;
}
```

Output:



```
Enter the number :
2
sum of proper divisors is
1
```



```
Enter the number :
20
sum of proper divisors is
22
```

Result:

Thus, the program to find the sum of proper divisors using Functions in C language has been executed and verified successfully.

B. DIFFERENCE IN LCM AND HCF.

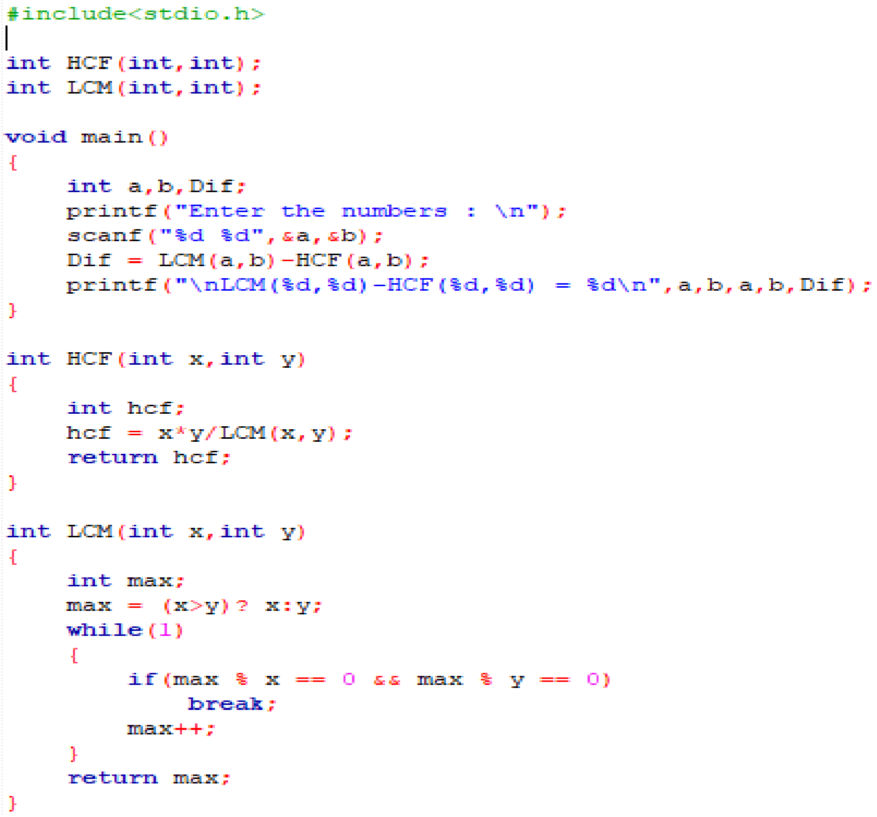
Aim:

To Write a program in C to Print Difference Between LCM and HCF of the given numbers using Functions.

Algorithm:

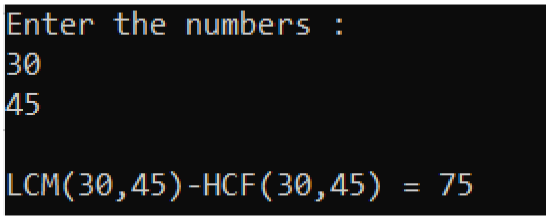
```
MAIN:
    Step 1: Start
    Step 2: Declare three integral variables num1, num2, dif and get the values of num1 and num2.
    Step 3: Calculate dif = LCM (num1, num2) – HCF (num1, num2).
    Step 4: Print the value of dif.
    Step 5: Stop

LCM (x, y):
    Step 1: Start
    Step 2: Declare three integral variables max.
    Step 3: calculate (x>y)? x: y and store to max.
    Step 4: While (1)
                If (max % x == 0 && max % y == 0)
                    Break;
                max++;
            End While.
    Step 5: Return max;

HCF (x, y):
    Step 1: Start
    Step 2: Declare three integral variables hcf.
    Step 3: Calculate x*y/LCM (x, y) and store to hcf.
    Step 4: Return hcf;
```

Program:

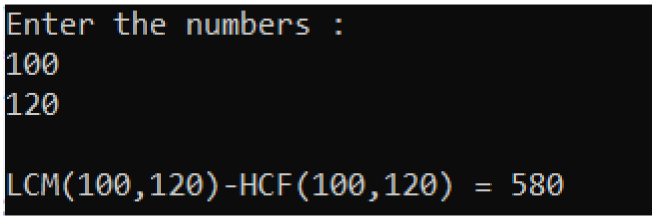


```c
#include<stdio.h>

int HCF(int,int);
int LCM(int,int);

void main()
{
    int a,b,Dif;
    printf("Enter the numbers : \n");
    scanf("%d %d",&a,&b);
    Dif = LCM(a,b)-HCF(a,b);
    printf("\nLCM(%d,%d)-HCF(%d,%d)  = %d\n",a,b,a,b,Dif);
}

int HCF(int x,int y)
{
    int hcf;
    hcf = x*y/LCM(x,y);
    return hcf;
}

int LCM(int x,int y)
{
    int max;
    max = (x>y)? x:y;
    while(1)
    {
        if(max % x == 0 && max % y == 0)
            break;
        max++;
    }
    return max;
}
```

Output:



```
Enter the numbers :
30
45

LCM(30,45)-HCF(30,45) = 75
```



```
Enter the numbers :
100
120

LCM(100,120)-HCF(100,120) = 580
```

Result:

Thus, the program to find Difference Between LCM and HCF of the given numbers using Functions in C language has been executed and verified successfully.

C. FRIENDLY NUMBERS OR NOT.

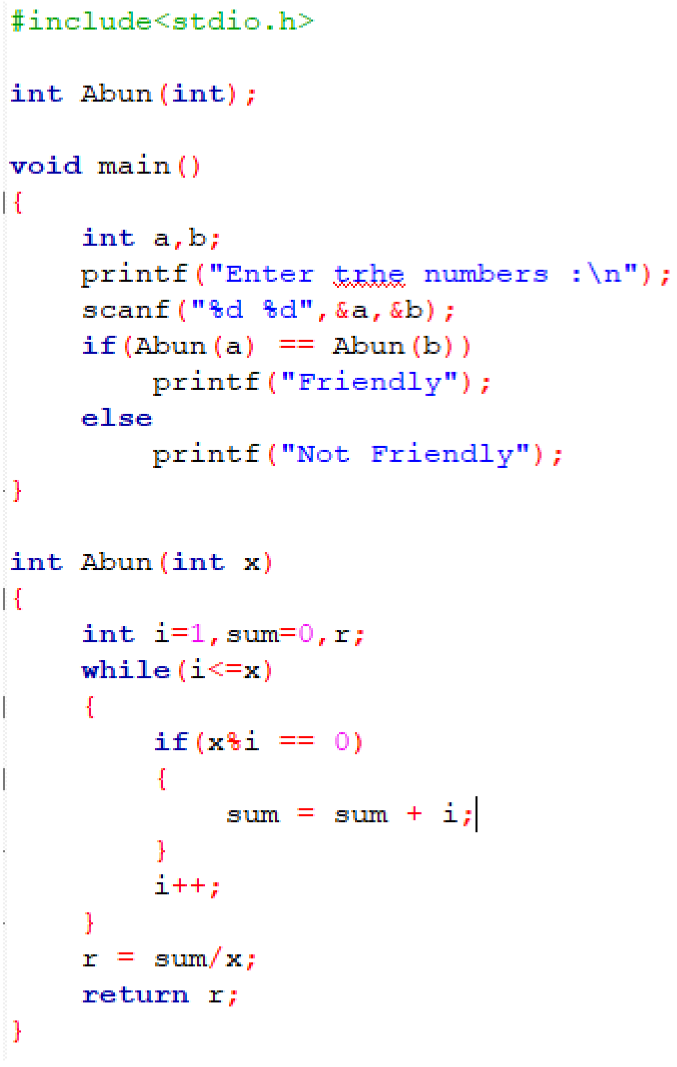
Aim:

To Write a program in C to Print the given numbers are Friendly or not using Functions.

Algorithm:

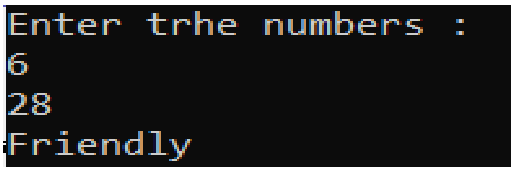
```
MAIN:
  Step 1: Start
  Step 2: Declare three integral variables num1, num2 and get the values of num1 and num2.
  Step 3: if (Abun (num1) == Abun(num2))
  Step 4: Print given numbers are friendly.
  Step 5: else Print Not friendly.
  Step 5: Stop

Abun(x):
  Step 1: Start
  Step 2: Declare int variable sum, r and initialize to zero
  Step 3: For (int i=1; i<x; i++)
              If (x%i == 0)
                  Sum = sum;
  Step 4: calculate sum/x and store to r
  Step 4: Return r.
```

Program:

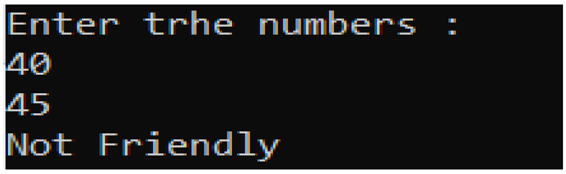


```c
#include<stdio.h>

int Abun(int);

void main()
{
    int a,b;
    printf("Enter trhe numbers :\n");
    scanf("%d %d",&a,&b);
    if(Abun(a)  == Abun(b))
        printf("Friendly");
    else
        printf("Not Friendly");
}

int Abun(int x)
{
    int i=1,sum=0,r;
    while(i<=x)
    {
        if(x%i == 0)
        {
            sum = sum + i;
        }
        i++;
    }
    r = sum/x;
    return r;
}
```

Output:



```
Enter trhe numbers :
6
28
Friendly
```



```
Enter trhe numbers :
40
45
Not Friendly
```

Result:

Thus, the program to find the given numbers are friendly or not friendly using Functions in C language has been executed and verified successfully.

D. AMICABLE OR NOT.

Aim:

To Write a program in C to Print the given numbers are Amicable or not using Functions.

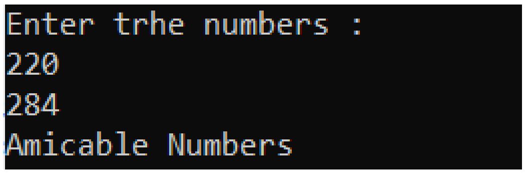
Algorithm:

```
MAIN:
  Step 1: Start
  Step 2: Declare three integral variables num1, num2 and get the values of num1 and num2.
  Step 3: if (SOPD (num1) == num2 && SPOD (num2) == num1)
  Step 4: Print given numbers are Amicable.
  Step 5: else Print Not Amicable numbers.
  Step 5: Stop
SOPD (x):
  Step 1: Start
  Step 2: Declare int variable sum and initialize to zero
  Step 3: For (int i=1; i<x; i++)
              If (x%i == 0)
                  Sum = sum;
  Step 4: Return sum.
```

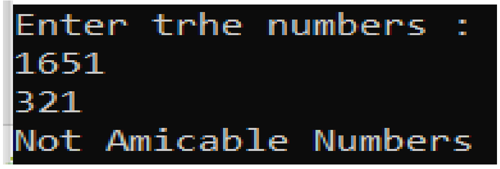
Program:

```c
#include<stdio.h>
int SOPD(int);
void main()
{
    int a,b;
    printf("Enter trhe numbers :\n");
    scanf("%d %d",&a,&b);
    if(SOPD(a) == b && SOPD(b) == a)
        printf("Amicable Numbers");
    else
        printf("Not Amicable Numbers");
}

int SOPD(int x)
{
    int sum=0;
    for(int i=1;i<x;i++)
    {
        if(x%i == 0)
        {
            sum = sum +i;
        }
    }
    return sum;
}
```

Output:



```
Enter trhe numbers :
220
284
Amicable Numbers
```



```
Enter trhe numbers :
1651
321
Not Amicable Numbers
```

Result:

Thus, the program to find the given numbers are Amicable or not Amicable using Functions in C language has been executed and verified successfully.

E. SUM OF SERIES.

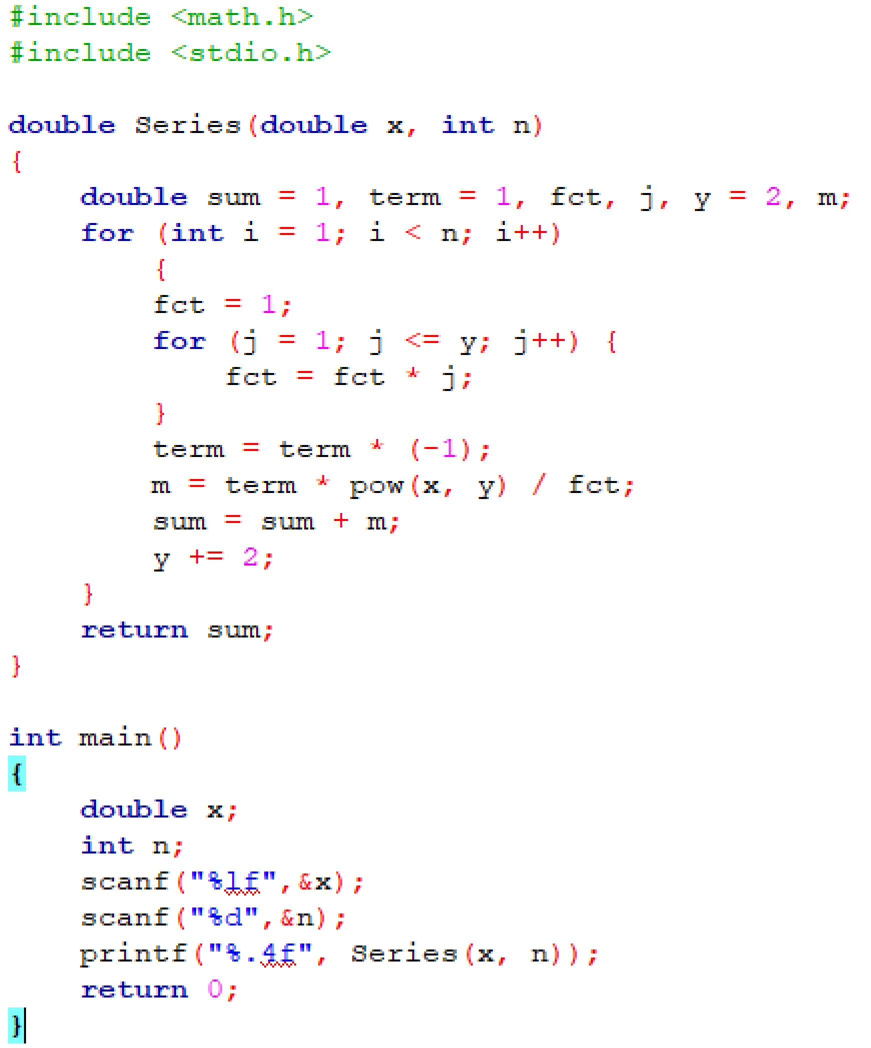
Aim:

To Write a program in C to Print the sum of $1-\frac{x^2}{2!}+\frac{x^4}{4!}-$ ... ... series using Functions.

Algorithm:

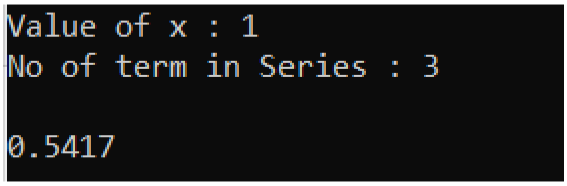
```
MAIN:
    Step 1: Start
    Step 2: Declare a double variable x and integer n (no of terms). And get the values
    Step 3: Call the function Series and print the value of sum.
    Step 4: Stop.
Series (x, n)
    Step 1: Start
    Step 2: Declare double variables sum = 2, term, fct =1, j, y =2, m;
    Step 3: for (int i = 1; i < n; i++)
                {for (j = 1; j <= y; j++)
                {fct = fct * j;}
                 term = term * (-1);
                 m = term * pow(x, y) / fct;
                 sum = sum + m;
                 y += 2;}
    Step 4: Return sum.
```

Program:

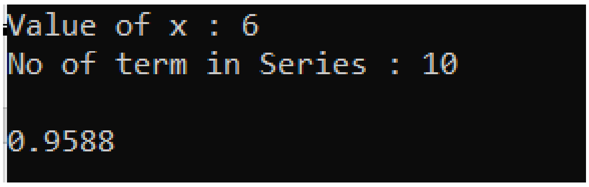


```c
#include <math.h>
#include <stdio.h>

double Series(double x, int n)
{
    double sum = 1, term = 1, fct, j, y = 2, m;
    for (int i = 1; i < n; i++)
        {
        fct = 1;
        for (j = 1; j <= y; j++) {
            fct = fct * j;
        }
        term = term * (-1);
        m = term * pow(x, y) / fct;
        sum = sum + m;
        y += 2;
    }
    return sum;
}

int main()
{
    double x;
    int n;
    scanf("%lf",&x);
    scanf("%d",&n);
    printf("%.4f", Series(x, n));
    return 0;
}
```

Output:



```
Value of x : 1
No of term in Series : 3

0.5417
```



```
Value of x : 6
No of term in Series : 10

0.9588
```

Result:

Thus the program to find the sum of the given series using Functions in C language has been executed and verified successfully.