CSE115L – Programming Language I Lab
Lab - 11
Functions

In this lab, we will solve a few problems using functions. The following examples will help you remember the syntax.

General syntax of writing functions

```
return_type function_name (parameters)
{
      local variable declaration;
      executable statement1 ;
      executable statement2 ;
      . . . . . .
      return statement;
}
```

Example 1: Write a function that prints the average of two integer values

```
#include <stdio.h>

void Average(int first, int second);

int main()
{
    int a = 7, b = 8;
    Average(a,b); // calling the function
    return 0;
}

void Average(int first, int second)
{
    printf("%f", (first+second)/2.0);
}
```

Example 2: Write a function that returns 1 if the integer passed to the function is a prime, and returns 0 otherwise

```
#include <stdio.h>

int is_prime(int a);

int main()
{
    int n;

    printf("Enter an integer: ");
    scanf("%d", &n);

    if(is_prime(n) == 1)
        printf("%d is a prime.", n);
    else printf("%d is not a prime.", n);
}

int is_prime(int a)
{
    int i;
    for(i=2; i<a; i++)
    {
        if(a%i == 0)
            return 0;
    }
    return 1;
}
```

Example 3: Write a function that returns the factorial value of the integer passed to it

```
#include <stdio.h>

int factorial(int x);

int main()
{
     int n;
     int result;
     printf("Enter an integer\n");
     scanf("%d",&n);
     result = factorial(n);
     printf("%d! = %d",n,result);
     return 0;
}

int factorial(int x)
{
     int fact = 1, i;
     for(i=1;i<=x;i++)
         fact *= i;
     return fact;
}
```

Some useful library functions

Function	Header	Purpose	Argument(s)	Result
`abs(x)`	`<stdlib.h>`	Returns the absolute value of its integer arguments	int	int
`ceil(x)`	`<math.h>`	Returns the smallest integral value that is not less that x	double	double
`pow(x,y)`	`<math.h>`	Returns x raised to the power of y	double	double
`cos(x)`	`<math.h>`	Returns the cosine of angle x	Double(radians)	Double
`sqrt(x)`	`<math.h>`	Returns the non-negative square root of x for x >= 0.0	Double	Double

Example 4: C program to compute the integer resulting from rounding a number n (using function)

```
#include <stdio.h>
int round1(float n)
{
    int i=n;  //integer part of n
    if(n-i>=0.5) return i+1;
    else return i;
}
```

```
int main()
{
    float n;
    printf("\nEnter a number: ");
    scanf("%f", &n);

    int s = round1(n);
    printf("%d",s)
}
```

Perform the following tasks.

Task 1: A perfect number is a positive integer that is equal to the sum of its proper positive divisors, that is, the sum of its positive divisors excluding the number itself. For example, 28 is a perfect number, since its proper positive divisors are 1, 2, 4, 7 and 14 (excluding 28) and 1+2+4+7+14=28. Write a function that accepts an integer parameter **n** and returns 1 if n is a perfect number, and returns 0 otherwise.

Task 2: Write a function that accepts two integer parameters **n** and **r**, and returns the value of $\boldsymbol{n_{C_r}}$.

Task 3: Write a function that accepts an integer **n** and returns the value $\boldsymbol{n_{C_1} + n_{C_2} + n_{C_3} + \cdots + n_{C_n}}$