RECURSIVE FUNCTIONS

Introduction

- Recursion is a process by which a function calls itself repeatedly, until some specified condition has been satisfied.
- The process is used for repetitive computations in which each action is stated in terms of a previous result.

Introduction

- In order to solve a problem recursively, two conditions must be satisfied.
- First, the problem must be written in a recursive form, and second, the problem statement must include a stopping condition.

Recursion - Example

- Suppose, for example, we wish to calculate the factorial of a positive integer quantity.
- We would normally express this problem as:

$$n! = 1 \times 2 \times 3 \times \ldots \times n$$

  - where **n** is the specified positive integer.

Recursion - Example

- However, we can also express this problem in another way by writing: $n! = n \times (n - 1)!$
- This is a recursive statement of the problem, in which the desired action (**the calculation of n!**) is expressed in terms of a previous result [the value of (n-1)!, which is assumed to be known].

Recursion - Example

- Also, we know that 1! = 1, by definition.
- *This last expression provides a stopping condition for the recursion.*

Calculating Factorials - Example

```
# include <stdio.h>
long int factorial ( int n) {
int i;
long int prod = 1;
if ( n > 1)
    for ( i = 2; i<=n ; ++i)
        prod *= i;
        return(prod);      }
void main ( void )    {
printf ( "\n 5! = %ld", factorial(5));
                          }
```

```
# include <stdio.h>
long int factorial ( int n) {
if ( n <= 1)
    return 1;
else
    return ( n * factorial ( n – 1 ));
                          }
void main ( void )
    {
printf ( "\n 5! = %ld", factorial(5));
    }
```

Practice Questions

- Convert the following function into recursive function:

```
int sum(int n)
{
  int i, s = 0;
  for(i=1;i<=n;i++)
        s += i;
  return s;
}
```

Practice Questions

- Write a C function for the following mathematical formulations:
  - F(x) = $\begin{cases} n.(n-2), & n \text{ is positive and even} \\ n.(n-1), & n \text{ is positive and odd} \\ 1, & n < 0 \end{cases}$

- Write a function to print the n^{th} term of Fibonacci series.

- Write a void returning recursive function to convert the given decimal no. system integer into its corresponding binary integer.