Problem 1. Recursive Conversion

Convert the following function to one that uses recursion.

```
void sign(int n)
{
   while (n>0)
      count << "No Parking\n";
   n--;
}
```

Problem 2. Recursive Array Sum

Write a function that accepts an array of integers and a number indicating the number of elements as arguments. The function should recursively calculate the sum of all the numbers on the array.

Problem 3. Recursive Multiplication

Write a recursive function that accepts two arguments into the parameters x and y. The function should return the value of x times y. Remember, multiplication can be performed as repeated addition:

```
7 * 4 = 4 + 4 + 4 + 4 + 4 + 4 + 4
```

Problem 4. Sum of Numbers

Write a function that accepts an integer argument and returns the sum of all the integers from 1 up to the number passed as an argument. For example, if 50 is passed as an argument, the function will return the sum of 1, 2, 3, 4, … 50. Use recursion to calculate the sum. Demonstrate the function in a program.

Problem 5.

The formula for computing the number of ways of choosing r different things from a set of n things is the following:

$$C(n,r) = \frac{n!}{r! * (n-r)!}$$

The factorial function n! is defined by

$$n! = n * (n-1) * (n-2) * \ldots * 1$$

Discover a recursive version of this formula and write a recursive function that computes the value of the formula. Embed the function in a program and test it.

Problem 6.

Write a recursive function that has an argument that is an array of characters and two arguments that are bounds on array indexes. The function should reverse the order of those entries in the array whose indexes are between the two bounds. For example, if the array is

```
a[0] == 'A' a[1] == 'B' a[2] == 'C' a[3] == 'D' a[4] == 'E'
```

and the bounds are 1 and 4, then after the function is run the array elements should be

```
a[0] == 'A' a[1] == 'E' a[2] == 'D' a[3] == 'C' a[4] == 'B'
```

Embed the function in a program and test it. After you have fully debugged this function, define another function that takes a single argument which is an array that contains a string value and that reverses the spelling of the string value in the array argument. This function will include a call to the recursive definition you did for the first part of this project. Embed this second function in a program and test it.