

1. (5 pts) Give the exact output produced by the following program

```
#include <iostream>
using namespace std;
```

```
void func( int x, int &y );
```

```
// Output here :
```

```
int main()
{
    double x = 3.14;
    int    y = 13;
    int    z = 2;

    func( x, y );
    cout << "main: " << x
         << "-"      << y
         << "-"      << z << endl;

    return 0;
}
```

```
func: 4-14
main: 3.14-14-2
```

```
void func( int x, int &y )
{
    int z;
    x = x + 1;
    y = y + 1;
    cout << "func: " << x
         << "-"      << y << endl;
    z = x + y;
}
```

2. (6 pts) Here's a prototype:

```
void rep( char ch, int count );
```

When called, the function prints the given character 'count' number of times.

For instance, the function call `rep('x', 7);` would print this to the screen:

```
xxxxxxx
```

Write a **RECURSIVE** definition of `rep()`, using the same prototype as above.

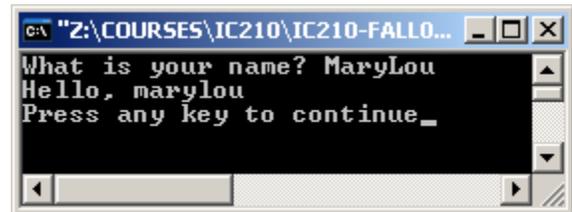
```
void rep( char ch, int count )
{
    if( count == 0 ) return;
    cout << ch;
    rep( ch, count-1 );
}
```

3. (4 pts) Consider a function named **makeLowercase** with one parameter, a C++ string. The function returns a C++ string identical to the parameter, except that all uppercase letters have been changed to lowercase.

a.) (2 pts) Write the prototype for this function:

```
string makeLowercase( string s );
```

b.) (2 pts) Assume function **makeLowercase** has been correctly defined. Complete the code fragment below such that it prints **Hello**, then prints the entered name in all lowercase letters. You must use function **makeLowercase**. You may make any additional declarations you think you need.



```
string name;  
cout << "What is your name?";  
cin >> name;
```

```
cout << "Hello, " << makeLowercase( name );
```

4. (11 pts) Write a function named **arrayOfSquares** with one parameter only: an integer N that indicates the size of an array to be created dynamically. Your function will create an array of that size and fill it with the first N integer squares: $1^2 \dots N^2$. It will then 'return' the array to the calling function. For instance, calling `arrayOfSquares` with size 5 would create this array:

| | | | | |
|---|---|---|----|----|
| 1 | 4 | 9 | 16 | 25 |
|---|---|---|----|----|

(a) (2 pts.) Give the prototype for the function **arrayOfSquares**.

```
int* arrayOfSquares( int N );
```

(b) (6 pts.) Write the definition for function **arrayOfSquares**.

```
int* arrayOfSquares( int N )
{
    int *A = new int[N];
    for( int i = 1; i <= N; i++ )
        A[i-1] = (i*i);
    return A;
}
```

(c) (3 pts.) Give the code fragment, as it would appear in main, to use **arrayOfSquares** to create an array of the first 5 squares, **then output the first and last values in the array, separated by a space**. Include all necessary variable declarations.

```
int* a = arrayOfSquares( 5 );
cout << a[0] << ' ' << a[4] << endl;
```

5. Provide the *prototypes* (**not** the definitions) for the following functions.

a. (4 pts) Function name: **makeArray**. Description: Takes an integer N and creates an array of N integers, where each value is initialized to zero.

Example use:

```
int N = 10;
int *array;
makeArray( N, array );

for( int i = 0; i < N; i= i + 1 )
    cin >> array[i];
```

Prototype: void makeArray(int N, int* &A);

Note: (a) This will **not** work: void makeArray(int N, int& *A);

(b) This will **not** work: void makeArray(int N, int& A[]);

b. (4 pts) Function name: **minValue**. Description: Returns the minimum value in an array.

Example use:

```
int size = 10;
double *anArray = new double[size];
for( int i = 0; i < size; i = i + 1 )
    cin >> anArray[i];

cout << "The array minimum is "
     << minValue( anArray, size ) << endl;
```

Prototype: double minValue(double A[], int N);

- or - **double minValue(double* A, int N);**

c. (4 pts) Function name: **printWords**. Description: Prints a word N times, either to a file or to the screen.

Example use:

```
ofstream fout("file.txt");
printWords( "hello cout!", 7, cout );
printWords( "hello file!", 7, fout );
```

Prototype: void printWords(string s, int N, ostream& os);

6. (12 pts.) A function named **countChars** takes 4 arguments, in this order:
- an array of characters
 - an argument named **N** indicating the size of the array
 - an argument named **letterToCount** indicating some letter to be counted
 - an argument named **firstIndex** which, after executing the function, will contain the index in the array where **letterToCount** is first found. If the array does not contain **letterToCount**, then **firstIndex** should be set to -1.

The function will scan the array and count how many times **letterToCount** is found, then return this count. For instance, if **letterToCount** was 'b' and the following array passed to the function:

| | | | | |
|-----|-----|-----|-----|-----|
| 'b' | 'c' | 'd' | 'b' | 'b' |
|-----|-----|-----|-----|-----|

then the function would return 3 and **firstIndex** would be set to 0.

- (a) (4 pts.) Give the prototype for the function **countChars**.

```
int countChars( char c[], int N, char letterToCount, int& firstIndex );
```

- (8 pts.) Write the definition of function **countChars**.

```
int countChars( char c[], int N, char letterToCount, int& firstIndex )
{
    firstIndex = -1;
    int count = 0;
    for( int i = 0; i < N; i++ )
        if( c[i] == letterToCount )
        {
            count++;
            if( firstIndex == -1 ) // i.e., this is the first time we have
                firstIndex = i;    // encountered the letter we're counting
        }
    return count;
}
```