CSC112 Fall 2010
Fundamentals of Computer Science
Practice Problems

Note: Some of these problem are too long (time-wise) to include on a 50-minute exam. Also, there are too many problems here to complete in a 50-minute exam. But, they review the material, and help prepare you for the exam.

1. Write a C++ function to accept an array of characters (C-style character strings, terminated by the null character `'\0'`) and return the length of the string. *Hint:* Count the number of characters up to the character `'\0'`. The function header should be:

   ```
   int string_length( char * s )
   ```

   Try this exercise again, with the function header:

   ```
   int string_length( char s[] )
   ```

2. Describe how to embed a 2-D array of type `float` into a 1-D array.

   (a) How would you declare the array ?

   (b) How would allocate memory for an array of m rows and n columns ?

   (c) How would you access the element in row i and column j ?

3. Describe how to implement a 2-D array using a pointer to a pointer. Suppose the declaration of an array is:

   ```
   int m, n       ;
   float ** table ;
   ```

   Write a segment of code to allocate memory for a 2-D array with m rows and n columns.

4. (**Slightly harder problem**) Write a C++ function to accept two arrays of characters (C-style character strings), `s` and `t`. Find the first occurrence of substring `t` in string `s`. Your function should return a (char) pointer to where the match occurs in string `s`. If no match is found, return the NULL pointer. The function header should be:

   ```
   char * find_string( char *s, char * t )
   ```

5. Write a C++ function to accept the name of a file containing binary integers, and compute the sum of the numbers in the file. The function header should be:

   ```
   int file_sum( char * fname )
   ```

6. Write a C++ function to accept the name of a text file and count the number of sentences in that file. Assume every sentence ends with a period, and there are no other uses of the period character in the file. I.e., read the file one character at a time and count the number of periods. The function header should be:

```
int count_periods( char * fname )
```

7. Give the declaration of a C++ structure (named `struct polynomial`). to implement a polynomial. You may assume that any polynomial that we will use is of degree less than 40. An example of a polynomial is given by:

$$p(x) = 2 + x - 3x^2 + 5x^3 + x^4 - 2x^5$$

Hint: A polynomial can be represented by a list of its coefficients.

8. Using your declaration from problem 7, write a C++ function to evaluate the polynomial at x. The function header should be:

```
double polynomial_eval(struct polynomial p, double x)
```

9. Describe the following features of C++ classes:
   (a) Constructor
   (b) Destructor
   (c) Private data
   (d) Public data
   (e) Friend functions
   (f) Scope resolution operator **::**

10. What is the purpose of private data as a language feature in C++ ? I.e., why would the language designers include this feature ?

11. Suppose we want to design a C++ class to represent a list of numbers. I tried the following declaration to do this:

```
class num_list {
  public:
   const int max = 100 ;
   int list[max] ;
   int howmany   ;
} ;
```

When trying to compile this (file name: c.cc), I get the following error message:

```
c.cc:3: error: ISO C++ forbids initialization of member max
```

Why do I get this error message ? If I change the declaration to:

```
class num_list {
  public:
   static const int max = 100 ;
   int list[max] ;
   int howmany   ;
} ;
```

the class compiles with no error. Why ?

12. Write the declaration of a C++ class which is intended to represent a list of names and phone numbers. Include public member functions for the operations insert, delete, and find. Just write the declaration of the class, but not the implementation.

13. Below is C++ code to implement a sorting method known as *insertion sort*. How could you modify the code to sort a list of names instead of a list of numbers ?

```
// ------- function insertion_sort() ----------------------------------
// Performs insertion sort.
void insertion_sort( int a[], int n )
{
    int i, j, temp ;

    // Start by considering the second element (in position a[1]).
    // Loop over each subsequent element in the array and insert
    // it into the correct position in the portion of the array
    // to the left.  Once this is done, the array is sorted up to
    // and including position i.
    for ( i = 1 ; i < n ; i++ ) {
        //  Here, the array is currently sorted up to position i-1.
        //  Next, we arrange things so it is sorted up to position i.

        temp = a[i] ;  // Save the value in position i into a temporary.
        j = i - 1 ;    // Start looking left (position i-1).

        // Find the position in the array where a[i] belongs.
        while ( ( j >= 0 ) && ( temp < a[j] ) ) {
            a[j+1] = a[j] ;  // Move the element in position j forward.
            j-- ;            // Decrement j (go left) to prepare for the
        }                    // next iteration of the loop.

        // At this point, we have completed the loop.
        // Either j == -1, or we have found a spot (index j)
        // for which temp >= a[j].   That means that the value
        // held in 'temp' belongs immediately to the right of
        // position j.
        a[j+1] = temp ;
    }
}
```