Name: _______Answers___________

CS 173 Opportunity

Directions:

1. Declare an array called student and write C++ code to read student names from a file called “studentnames.txt” and put the names into the array.

```
const int size = 100;
string student [size];
ifstream infile;
string name;
infile.open(“studentnames.txt”);
for (int i = 0; i < size; i++)
{
        infile >> name;
        student [i] = name;
}
```

2. How does C++ support abstract data types (ADT’s) so that programmers can create new types? Be sure to mention public and private in your answer. Explain the role of templates.

A programmer can create a new type by using a class. The correct level of abstraction is achieved by putting the data in the private section. The functions that users of the new type can access go in the public section. Clients can declare variable of the new type and can manipulate them in their programs that include the given class. Template allow the entries into container types to be declared by the client, thereby making it possible to use one set of code for many types.

3. Show how the compiler might use memory to accommodate the following code:
   a. int *x, y, *z = new int;
      y = 1;
      x = &y;
      *z = 3;
      // Draw a memory picture for the state at this point. Assume that the memory locations are labeled starting at 0 and each location contains a byte. Assume that the Operating System assigns y to be at location 7.

      Note: z points to a location that contains 3. z itself has a value at some other location. In my example, I put z at location 2 showing that z points to location 4 which has the value 3. I put x at location 9 and show that x contains the address of y which is 7. I put the addresses along the left side of the table.

0	
1	
2	z = 4
3	
4	3
5	
6	
7	y = 1
8	
9	x = 7
10	
11	

   b. x = z;
      *x = 5;
      y = *x + *z;
      Note: the pointer x now has the same value as z, which is location 4. The location that x points to (location 4) gets the value 5. Now both x and z point to the same location and so when we add their contents, we get 10.
      // Draw a separate memory picture for the state at this point

0	
1	
2	
3	
4	5
5	
6	
7	y = 10
8	
9	x = 4
10	
11	

4. The following code is supposed to perform a selection sort on an array called A that has 128 string entries. However, there may be some errors in the code. Find 4 errors and correct them. Tell what the big O value is for the correct sort.

```
void sort (string A [n])   need a number for n
{
    int start, , tempIndex;
    string min;
    for (start = 0; start <   n; start = start + 1)
            {
                    tempIndex = start;
                    min = A[start];
                    for ( int i = start + 1; i <  n; i++ )
                    {
                            if ( A[i] < min )
                            {
                                    min = A[i];
                                    tempIndex = i;
                            }
                    }
                    A[tempIndex] = A[start];
                    A[start] = min;
            }
    }
```

O(n^2)

5. Write pseudo code to perform a binary search on a sorted array called A that has 128 entries. Tell what the big O value is for the search.

See page 601 in your textbook.

O(log n)

6. Suppose you are given a list.h file as follows:

```
#include <iostream>
using namespace std;
template <class T>
class List_Position
{
        private:
                struct ListPart
                {
                        T data;
                        ListPart*  next;
                        ListPart( T value, ListPart nodeptr = NULL)
                        {
                                data = value;
                                next = nodeptr;
                        }
                }
                ListPart<T>* front;
                ListPart<T>* current;
                ListPart<T>* previous;

        public:
                List_Postion( );
                ~List_Postion( );
                void insert ( T );
                void advance ( );
                T remove( );
                int Length_of_Remainder ( );
                void reset( );
                void swap_remainders (List_Position L):
};
```

Write the code for List_Position.cpp for the insert function.

```
template <class T>
void List_Position<T> :: insert ( T value)
{
        ListPart<T>* L = new ListPart<T>(value);
        if (current == front)
        {
                L -> next = current -> next;
                current = L;
                front = L;
        }
        else
        {
                L -> next = current;
                current = L;
                previous-> next = L;
        }
}
```