**CSCI 2170 Review for test 2 (Wednesday, March 16th 2011 )**

**Class (chapter 3)**

- Be able to define a class given a problem specification
- Constructors
- Copy constructor
  - Shallow copy vs. deep copy
- Overloaded function
- Overloaded operator (==, !=, <=, =, >=, <<, >>)
- Destructor
  - When is it necessary to define a destructor for a class?
  - When is destructor called in a program?

**Recursion (chapters 2 and 5)**

- Four basic components for each recursive function
- Trace the execution (including backtracking) with recursive function
- Write recursive function for simple problems

**C++ pointers (chapter 4)**

- Understand the difference between static and dynamic allocation of memory for variables
- Understand the meaning and how to use pointer related operators, such as ***, &, →, ., new, delete**
- Be able to set up and use pointers to
  - simple data types (int, float, char, …)
  - struct data types
  - 1D and 2D array
- Know how to pass pointer variable to function

**Example questions:**

1. Questions in the homeworks
2. Questions in the textbook:
   - Page 115-117: Exercise 3, 4,5, 6, 11, 12, 14, 15
   - Page 167-169: Exercise 1, 2, 3, 7, 8, 11, 12

3. An ADT sorted list is used to build and maintain a list of sorted item. The members of sorted list class is similar to a list class, except for the insert and remove operations. Write the implementation of the insert and remove member functions:

```
void sortedlist::insert(listItemType& newitem, bool&success)
// this function will insert "newitem" into the list of items such that the items will remain sorted after the new item is inserted

void sortedlist::remove(listItemType item, bool&success)
// this function will remove item from the list, keeping the remaining items in sorted order
```

4. Show output from the following C++ code segment:

```
typedef   int * intPtr;
void  func(int *, int);
int  main()
{
      intPtr   p, q;
      int    x, y;

      p = & x;
      *p = 4;
       q = & y;
       y=6;
       cout << *p << "  " << *q << "  " << x  << " " << y << endl;
```

```
    q = p;
    *p = 3;
    cout << *p << "  " << *q << "  " << x  << " " << y << endl;

    func (&x, y);
    cout << *p << "  " << *q << "  "  << x  << " " << y << endl;

    p = new int;
    *p = *q+y;
    x = x + *q;
    cout << *p << "  " << *q << "  "  << x  << " " << y << endl;

    delete p;

    p=NULL;
    q=NULL;

    return 0;
}
void func(int * p1,  int p2)
{       p2++;
    *p1 = *p1+p2;
}
```

4. Given a structure defined as:

```
struct   CarStruct
{          string   maker;
           string   model;
           float    price;
};
```

(a) declare a pointer variable "pointer" that can be used to a CarStruct type data
(b) dynamically allocate space for a CarStruct type data, and let "pointer" points to that memory location
(c) assign 24000.00 to be price of the newly created structure of type CarStruct
(d) release the memory located in (b) and make "pointer" not pointing to anything
(e) modify the CarStruct structure to include one additional member "next" which is a pointer to CarStruct type data