CSCI 2170 Test 3 review

1. **Struct**
   a. Struct declaration
   b. Access struct members
   c. Pass struct variable to function
   d. Array of struct

2. **Pointer, dynamic memory allocation**
   - How to declare pointer variable
   - Use pointer variable to point to memory locations
   - Use new operator to acquire memory during run time
   - Use delete operator to release memory

3. **Linked list related questions**
   - Define a node structure for linked list
   - Create a new node using dynamic memory allocation
   - Free the memory of a node
   - List traversal:
     - Print all the values stored in a linked list
     - Compute the sum of all the values in a linked list
     - Find whether a value is in the list
     - Update a value in the linked list to a new value

     Insertion operation
     - Insertion always happen at the front of the list
     - Insertion always happen at the end of the list
     - Insertion by location
     - Insertion into sorted list and maintain the list to be sorted after insertion

     Deletion operation
     - Deletion by location
     - Deletion from a sorted list and maintain the list to be sorted after deletion

   - Build a list by read data from a data file
   - Destroy a list

4. **Abstract Data Type (ADT) related questions:**
   - Be able to define a ADT by defining its header file and specification file
   - Be able to write client program using the ADT defined above
   - Be able to write a client program using ADT list (array implementation or linked list implementation)
   - Understand how the array based list ADT is defined.
     - Be able to add new methods by defining the method in the header file and define the method in the implementation file
   - Understand how the linked list based list ADT is defined
     - Be able to add new methods by defining the method in the header file and define the method in the implementation file
   - **Be able to write client program using the ADT list**

EXAMPLE QUESTION:

1. What is the output of the following program segment? **Write your answer below each cout statement.**

```
typedef     int   *   intPtr;
intPtr        p, q;
int             x=5, y=10;

p=&y;
q=p;
*q = *p+10;
x=*q - 10;
y= x - 5;
cout << x << "  " << y << " " << *p  << " " << *q << endl;

p=new int;
*p = 5;
*q = *p + 5;
x = *q;
cout << x << "  " << y << " " << *p  << " " << *q << endl;
```

2. (2 pts) Show the C++ statement to release the memory space pointed at by the pointer 'p'.

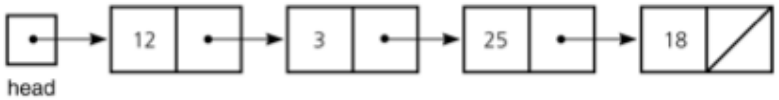
3. (2 pts) Suppose the following structure and variables are declared:

```
struct    Student {
      string      name;
      string      email;
   };
   Student        Peter;
   Student*       p = &Peter;
```

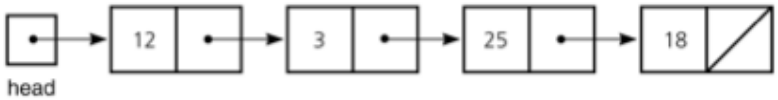
Which of the following statements assigns a new email to Peter?

(a) p[email] = "Peter@mtsu.edu";
(b) p→email = "Peter@mtsu.edu";
(c) p.email = "Peter@mtsu.edu";
(d) Peter[email] = "Peter@mtsu.edu";
(e) Peter→email = "Peter@mtsu.edu";

4. For each of the 4 questions a-d shown below, only the head of the **linked list** is given. Show the code for that <u>specific linked list</u> only. Do not need to write the code for a general problem.



a. Show C++ loop statements needed to print all the values in the linked list, one value per line.



b. Suppose pointer variable ***prev*** points to the first node (node with value 12) in the list and ***cur*** points the second node (node with value 3). Write the C++ statements to **remove the second node from the list and free its memory.**

1. Construct a linked list based on user inputs (from keyboard, or from file)
2. Show function to traverse the linked list (i.e., print data from all the nodes)
3. Show function to add data to a linked list
4. Show function to delete a value from a linked list

5. A complex number consists of two components: the real component and the imaginary component. An example of a complex number is 2+3i, where 2 is the real component and 3 is the imaginary component of the data. Define a class MyComplexClass. It has two data values of float type: **real** and **imaginary.**

This class has the following member functions:

- A **default constructor** that assigns 0.0 to both its real and imaginary data members;
- The **value constructor** that assigns client supplied (real and imaginary) values to the real and imaginary data members;
- The copy constructor
- Define the **accessor** and the **mutator** functions, for example
    - A **member function "SetValues"** that assigns client supplied values to the real and imaginary data members; (This is not a constructor);
    - A member function "**GetReal**" that returns the real component of the number;
- A member function "**Display**" that outputs the complex number in the form "a + bi" on screen, where a and b are the real and imaginary components.
- A **member function "EqualTo"** that compares two complex numbers. It returns true if they are the same, and returns false if they are different. Two complex numbers are considered the same if the real components of the two values are the same and the imaginary components of the two values are also the same.
- Call method to display the complex objects
- Overloaded < operator, overloaded == operator
- Write user defined functions that pass complex objects by value or by reference

<u>You are required to:</u>

(a) Write the complete header file for MyComplexClass;
(b) Write the complete implementation file for MyComplexClass.
(c) Write the client program to:

- Create two objects of MyComplexClass. One objects should be created using the default constructor, and the other with the value constructor;
- Use "Set" methods to change the first object to the complex number 5.5+3i
- Use "Display" method to display the first object
- Apply **EqualTo** function to compare the two complex numbers and output appropriate messages concerning whether the two numbers are the same or not.
- Declare the third complex number as a copy of the second complex number, ie., using copy constructor
- Write a user defined functions to
    - Add two complex numbers. For two complex numbers: a+bi and c + di, the addition of the two numbers is: (a+c) + (b+d)i
- Declare an array of 20 complex numbers
- Use overloaded operator to compare two MyComplexClass objects
- Write a user defined functions to
    - Assign the values of each of these 20 complex numbers
    - Display each of these 20 complex numbers