Practice questions

COMP-322, Winter 2012, All Sections

These questions are not necessarily the same as a final as they aren't necessarily exactly representative of the degree of difficulty, length, or material of the exam. That is, you should not assume that the questions on the quiz will be exactly like these.

List of important topics

Here is a rough list of the topics we've covered so far that would be examinable.

1. Pointers, pointers to struct, * and & operators, pointer arithmetic on arrays
2. Linked lists
3. References
4. Memory management, memory leaks. new/delete
5. Basics of iterators: how to iterate over a list, vector, etc. How to write a function to take as input iterators.
6. Basics of streams

Part 1 (0 points): Warm-up

Warm-up Question 1 (0 points)

Suppose I have the following functions:

```
void swap(int& a, int&b) {
    int temp = a;
    a = b;
    b = temp;
}

int main() {
    int x = 3;
    int y = 4;
    ?????
}
```

What would I write in place of the ????? if I wanted to call the function to properly swap x and y?

Rewrite swap() so that it took as input two `int*` instead of `int&`. How would that affect the way you call the method?

Warm-up Question 2 (0 points)

What is the problem with the following code?

```
struct Point2d
{
     int x;
     int y;
};

Point2d* makePoint(int x, int y) {
       Point2d point;
       point.x = x;
       point.y = y;
       return &point;
}
```

How could one solve this problem and still return a Point2d?

Warm-up Question 3 (0 points)

Suppose I have the following struct:

```
struct IntNode
{
     int data;
     IntNode* next;
}
```

Write a function that takes as input a pointer to an `IntNode` that represents a linked list. Assume that it is sorted in increasing order by the int data. Insert the new node in the correct location. You should obey proper memory management.

Warm-up Question 4 (0 points)

Suppose there exists a function `IntNode* fun(IntNode* x)` . Which of the following are valid? For those that are invalid, correct them. In cases that something prints, output what will print.

1.
   ```
   IntNode* bar = fun(NULL);
   ```

2.
   ```
   IntNode* bar;
   bar = &(*(fun(NULL)));
   ```

3.
   ```
   IntNode b = *(fun(NULL));
   ```

4.
   ```
   IntNode b;
   IntNode* c = *b;
   (*c) = b;
   ```

5.
   ```
   int x = 3;
   int&y = &x;
   cout << x;
   ```

6.
   ```
   int x = 3;
   int&y = x;
   y = 10;
   cout << x;
   ```

7.
   ```
   int x = 2;
   int* y = &x;
   y = new int;
   (*y) = 3;
   cout << x;
   ```

8.
```
int x[10];
for (int j = 0; j < 10; j++) {
x[j] = 0;
}

int* p = (x[3]);
*(p + 3) = 4;
cout << x[3] << " " << x[6] << endl;
```

After answering the normal question, how would you rewrite the above using a reference instead of a pointer?

Warm-up Question 5 (0 points)
Write a function that takes as input a `const list<int>&` (from the stl library) and iterates over the entire list, summing all of the values.

Warm-up Question 6 (0 points)
Write a function that takes as input a start iterator for a container of `int` and an end (one past the finish) of a forward iterator and sums all the values in the container.

Now write how you would call this function to take the sum of all the values in the `vector<int> v`

Warm-up Question 7 (0 points)
What is the advantage of using references? Why are they better than pointers in some cases? How does this relate to the notion of "l-value" (something you can put on the left side of an equation)

Warm-up Question 8 (0 points)
What is the difference between the * operator and the & operator?

Warm-up Question 9 (0 points)
What is wrong with the following code?

```
//adds an element to a list at the front if it isn't already in the linked list,
//returns the element (linked to original list) or original list depending
//on if it was found or not
IntNode* insertUnique(IntNode* root, int value)
{
     IntNode* newNode = new IntNode;
     newNode->value = value;
     newNode->next = NULL;

     for (IntNode* current = root; current != NULL; current = current->next)
     {
         if (current->value == value)
         {
                 return root;
         }
     }

     //insert at the beginning now
     newNode->next = root;
     return newNode;
}
```

Warm-up Question 10 (0 points)
Explain the difference between how the computer evaluates `cout` in C++ vs a more standard method

call such as `printf()` in C or `System.out.println()` in Java.

In other words, what is the difference between writing:

```
System.out.println("hi" + "bye");
OR
printf("hibye");
```

vs

```
cout << "hi" << "bye";
```