CSci1113 Spring 2018
Study Guide: Final Exam

This study guide has four parts:
- General comments and rules.
- A detailed list of topics.
- A few comments on study questions and other review resources.
- Sample questions

I. General Comments and Rules
Here is some important administrative information about the final exam:
- *The exam will be Tuesday, May 8 from 6:30pm - 8:30pm in* ***Tate 101 (normal room).***
- *Please bring a photo ID.* We might or might not check IDs, but make sure to bring yours.
- The exam will be open book and notes. Electronic devices are acceptable, but they may **ONLY** be used for viewing notes and **not** compiling or communicating with others.
- Although the exam is open book, there might not be enough exam time to consult it extensively. So you should know the material well enough that you do not need to look up many items.
- The exam will be comprehensive, although more recent material (classes, dynamic allocation and pointers) will be more prominent than other material.
- You should know the general idea of recent material that was covered in lecture but was not used in labs or the homework. However, you do not need to know this material in great depth, such as copy constructors, inheritance and late/early binding.
- Expect to choose 10 of 12 questions (some of which are multi-part) on different topics.

II. Topics
You are responsible for all the material presented in the course. This includes the reading assignments, lectures, lab exercises, and homework exercises. Test questions may be drawn from any of these, but are more likely to involve concepts or C++ constructs that have appeared often in the lectures, labs, and homework problems.

Below is a list of topics since Midterm 2 that *might* be on the exam. This list is not comprehensive, but includes the most likely recent topics. In addition to these topics, any of the earlier material is also fair game. (See the Midterm 2 Study Guides for lists of earlier topics.)
1. Class motivation
2. Class basics: member variables and member functions, private vs. public, object instantiation and use
3. Class files in C++: class interface (.hpp) and class implementation (.cpp) files along with the purpose of #ifndef, #define and #endif
4. Member functions, including constructors, accessors, mutators, and other member functions
5. Classes and functions (objects as parameters, returning an object, etc.)
6. Friend functions
7. Overloaded operators
8. Classes having arrays as member variables
9. Arrays of objects
10. Dynamic allocation basics: motivations, basic ideas, new and delete
11. Pointers and pointer operations

12. Dynamically allocated arrays, including allocation, use, and deletion
13. Functions, pointers, and dynamic allocation (pointer variables as parameters, allocating an array within a function, returning a pointer, etc.)
14. Destructors (and a general idea of their relationship to copy constructors, overloaded operator=)
15. Inheritance basics: base and derived classes, derived class syntax, derived class interface and implementation, etc.
16. Derived class constructors, redefined member variables, access to inherited member variables and functions, etc.
17. Derived class use: instantiating and using objects of derived classes
18. Polymorphism: use of virtualization and storing children in a parent's type

As usual, exam problems may ask you to do the following types of tasks:

1. Answer short questions about any of the topics above.
2. Trace the execution of a code segment.
3. Locate syntax and logic errors in given C++ code.
4. Modify given C++ code to solve a problem.
5. Write C++ code (ranging from one-line answers on short questions, to an entire function, class, and/or short program on more involved questions).
6. Solve a given problem and write the solution as a C++ function or program.

III. Sample Problems and Other Resource Comments

There are a number of study resources for the final exam. In addition to the textbook, sample code, and past assignments and labs, there are/will be the following:

- Part of the final lecture will be review.
- The Midterm 2 Study Guide has lists of topics plus study questions for earlier topics. Since the final is comprehensive, there will be some questions on the final that involve earlier topics.
- In the next section is a list of some study questions on recent topics.

As usual, study questions in lab, lecture, and the study question file give examples of possible types of questions on possible topics. However, they are just examples; while some final exam questions will be similar types of questions on similar topics, the final exam might also include other types of questions on other topics as well.

IV. Sample Problems and Other Resource Comments

All sample questions deal with receipts from grocery shopping. For our purpose, we will assume that receipts are simply a list of strings (names of good purchased) along with an int (the corresponding price of each good).

A. Write a Receipt class that uses partially filled arrays to store the data. This should include relevant member variables, along with a default constructor and add() function that adds a single good and price to the receipt (as an input).

B. Change part A to use dynamic memory management with an additional input of how many items in total there are to a non-default constructor.

C. Rather than treating the stings and ints as unrelated, create a CheckoutItem class that stores all relevant information. Then modify the Receipt class to only have an array of CheckoutItems.

D. Overload the << operator for the CheckoutItem class so you can more easily show what information it holds.

E. Suppose we had two Receipt object instances. How would we swap them and what issues might arise?

F. What if we had two pointers to the (dynamically created) Receipt instances, how would swapping them be different? Along with answering this question, write a segment of code that would actually do the swapping.

G. Write a function getItem() that takes a single int as input and returns the CheckoutItem in the array at that spot.

H. Write a function subList() that takes a single int “n” as input and returns a dynamically allocated array that contains only the first n elements in the member variable array.

I. Write a member function of Receipt called addItem() that takes as input a CheckoutItem which will be added to the next open spot in the array.

J. Use all of the above to make a simple program to create a Reciept with the following items: (Milk, 500), (Pasta, 300). Then use subList() to get only the “milk” item. Finally, delete all dynamic memory.