COMP 1406 B/M: Winter 2015 Final Exam Study Guide

The exam is Saturday April 11, 2:00pm, Field House (Athletics).

Our review session is Thursday April 9, 10:00am, ME 4499.

General Studying Tips

A good strategy is to create your own study notes, preferably on paper (manually writing will help you remember what you are thinking about better, and help you organize your thoughts). Here's one possible way to make these notes:

- Go through the course learning objectives, slides, tutorials, and in-class code examples and make a list of key concepts that you should understand.
- On a separate piece of paper for each concept, write the concept at the top of the page.
- For each concept:
    - Write a general description of what the concept is about. Try explaining it as if you were teaching someone who has never programmed before.
    - Organize all the key points on the topic in a way that summarizes them well. This exercise will help you process the ideas and understand them better.
    - If you are weaker on one of the concepts, use the textbook and the resources on cuLearn and write everything you can about it.
- Also look for key examples of how each concept is used in code. Write down a description of how it's used and/or actual code examples (what you write depends on how easily you could write the code that does these things without looking it up).

At a minimum, use both the slides (which are really just teaching props) and the required readings.

Learn from assignments and quizzes. Whether you got perfect or not, look at your feedback, and feel free to ask us for more.

Try redoing the quizzes and see if you can get the right answer.

Practice the assignment and tutorial problems (especially if you had a hard time with them when you first saw them). If you didn't finish the tutorials, try the parts you didn't get to. Practice writing code on paper, since this is a lot harder to do than you'd think. At the very least, **you need to be able to code simple problems without outside resources**.

A few general studying tips:

- Find ways to stay relaxed. High stress will make your studying time far less effective. (Don't leave it to the last minute!)
- Try to stop working on your notes before your normal bedtime the night before the exam (if not sooner). Get a good night's rest - this really does matter!
- If you have time, you can spend some time memorizing some of your notes.
- On the day of the exam, review your notes. By now you don't want to be still trying to understand the concepts or memorizing key points if possible.

What You Can Bring With You

Just your pens and pencils. You can have a calculator if you really want to, but you shouldn't need it.

Layout of the Exam

There are 40 questions totalling 70 marks, plus a bonus worth 3.

Multiple Choice (30 questions, 1 mark each)

These questions will range from conceptual to code reading. Concepts from both C++ and Java will be covered. Here are the central topics for each question:

1. Problem solving techniques
2. Problem solving techniques
3. Memory arrangement of data: arrays
4. Memory arrangement of data: classes/structs
5. Memory arrangement of data: static attributes
6. Heap/stack
7. Heap/stack
8. Procedural style of working with objects vs OOP style
9. Object Behaviour: overloading methods
10. Object Behaviour: constructors
11. Object Behaviour: this
12. Object Behaviour: equals
13. Object Behaviour: methods
14. Reference structures: linked list

15. Reference structures: linked list
16. Reference structures: graphs
17. Reference structures: graphs
18. Goals of class use: encapsulation
19. Goals of class use: information hiding
20. Goals of class use: reuse
21. Goals of class use: encapsulation
22. Abstract data types
23. Shallow and deep copying
24. Advanced OOP: inheritance
25. Advanced OOP: abstract
26. Advanced OOP: super
27. Advanced OOP: access modifiers
28. Event-driven programming
29. Event-driven programming
30. Recursion

Written Questions

This section will focus entirely on Java. Here is some info about the questions:

1. Written conceptual, goals of class use (4 marks)
2. Written conceptual, polymorphism (5 marks)
3. Code reading/drawing, hierarchies (3 marks)
4. Code reading/drawing, hierarchies and object memory diagrams (4 marks)
5. Code reading, recursion (4 marks)
6. Code writing (3 marks)
7. Code writing (4 marks)
8. Written conceptual, polymorphism (4 marks)
9. Code writing (5 marks)
10. Code writing (4 marks)

All code writing will be cummulative in the sense that you need to understand everything you know about object-oriented programming in Java, from basic constructors to advanced OOP.

Bonus

Read and take notes on Chapter 7 of Think Like a Programmer (Solving Problems With Code Reuse). The bonus will cover this topic.

General Notes

You will not be asked to write recursive code, C++ code, or any Processing code.

Be able to articulate concepts well in English.

Be sure to state any assumptions you make when answering the questions.

Be strategic rather than starting at the beginning and working your way through. Read all the questions first, then start answering the questions you are most confident about. (Maximize your marks!)

Succeeding in Second Year

You learned a lot in this course, and all of it is a gateway into your second year. If you haven't read (all of) Think Like A Programmer yet, I recommend doing so over the summer. You can also keep up your skills by trying the book's practice problems in Java and/or C++.

In particular, pay attention to Chapters 7 & 8. Chapter 7 touches on a lot of the areas you'll be studying in more depth, like data structures and algorithms and object-oriented design patterns. Chapter 8 sums up what it means to think like a programmer, and how to improve your skills. If you are here because you are passionate about writing high quality code to solve interesting problems, there is a lot of great advice for you in this chapter.

Bye!

It was a pleasure having you in class, and I hope we cross paths in the future.

Never forget that with hard work and dedication, each of you can succeed - it just might take a little longer for some things to click for you. So if you don't obtain the C- you need from this class, don't give up! Figure out what you can do better next time and try again.