CS 2150 Exam 1

You MUST write your name and e-mail ID on EACH page and bubble in your userid at the bottom of EACH page – including this page.

If you are still writing when "pens down" is called, your exam will be ripped up and not graded – even if you are still writing to fill in the bubble forms. So please do that first. Sorry to have to be strict on this.

Other than bubbling in your userid at the bottom, please do not write in the footer section of each page.

There are 8 pages to this exam – once the exam starts, please make sure you have all 8 pages.

Questions are worth 4, 6, or 8 points, depending on the question length. The first and last page are worth 2 points each – if you fill in the bubble footer, you get those points. The four point questions on this exam should not take more than a line or two to answer – **your answer should not exceed about 20 words.** There are 100 points of questions and 1 hour 45 minutes (105 minutes) to take the exam, which means you should spend one minute per question point – the other 5 minutes are to fill out the bubble footers.

If you do not bubble in a page, you will not receive credit for that page!

This exam is CLOSED text book, closed-notes, **closed-calculator**, closed-cell phone, closed-computer, closed-neighbor, etc. Questions are worth different amounts, so be sure to look over all the questions and plan your time accordingly. Please sign the honor pledge here:

_____

_____

_____

_____

There are 10 types of people in the world –
those that understand binary and those that don't.

(the bubble footer is automatically inserted in this space)

C/C++

1. [4 points] Why might we want to write a *method* template?

2. [4 points] What are the three properties of references that differentiate them from pointers?

3. [4 points] Give a motivating example of when we would want to use implicit construction (i.e. when we would *not* want to have the constructor be `explicit`).

4. [4 points] Explain when the copy constructor called versus when is the `operator=()` method is called.

Lists

A *jump list* is a doubly linked list data structure where the elements are kept in **sorted** order. However, nodes in a jump list have additional forward and backward links (the so-called 'jump pointers') that allow one to rapidly transverse the list – each of these 'special' forward and backward links skip many items in the list, allowing for faster element access.

We'll assume the list has *n* elements. Given an element at position *i* in the list, it still has the regular doubly linked list links to *i-1* and *i+1*. There are also jump links, one forward and (possibly) one backward.

The forward jump link for an element at position *i* is to the element at position $i + \lfloor \sqrt[3]{i} \rfloor$, where $\sqrt[3]{i}$ is the cube root of *i*. Thus, the element at position 27 has a forward jump link to position 30, since 3 is the cube root of 27. The element at position 80 will have a forward jump link to position 84, since 4 is the floor of the cube root of 80 ($\sqrt[3]{80} = 4.3088$). All nodes have forward jump links.

The backward jump link follows a different pattern, however. If the position has an integer cube root (i.e., is one of 1^3, 2^3, 3^3, 4^3, etc.), then it has a backward jump link to the previous cube root. Thus, the element at position 64 (4^3) has a backward jump link to the element at position 27 (3^3). Any position that does not have an integer cube root does not have a backward jump link – to go backwards from that position, you have to move forward to the next position that is an integer cube root, and then start following the backward jump links to move backwards.

5. [8 points] What is the (big-theta) running time of search, insert, and delete for a jump list? **Briefly** explain why in each case.

6. [8 points] What are the benefits of this data structure? What are the drawbacks?

Numbers

7. [4 points] Why do some computers, such as the Intel x86, represent numbers in little Endian format?

8. [4 points] Convert 324 in base 6 to base 9.

9. [8 points] Encode -170 as a 32-bit little Endian two's complement integer (note the negative sign on that number). Show you work for partial credit!

Arrays and Order of Growth

10. [4 points] How does C++ store 2-dimensional arrays in memory? This can be explained in English or diagrammatically.

11. [4 points] How are command-line parameters accessed in C++? Give both the `main()` prototype, and BRIEFLY explain how the parameter(s) to `main()` are used.

12. [4 points] List everything you know about little-theta.

13. [4 points] Describe what must be done to show that a function *f* is big-O(*g*) – meaning, what is the formal proof procedure?

Floating Point Types

We have studied IEEE standard 754 for single-precision (i.e. 32 bit) floating point numbers. For the following three questions, you are going to encode the number in a different floating point number format that uses 3 bytes (i.e. 24 bits) instead of the 4 bytes (32 bits) that we have studied. You will be designing this data type, and then encoding a number into that data type. There are multiple right ways to design this data type, and your encoding will be graded independently of your type design. We will assume everything in this question is big-Endian.

14. [6 points] First, we must design the type. There will be one bit for the sign, of course. How many bits in the exponent? How many bits in the mantissa? Briefly, why did you pick those numbers?

15. [8 points] Encode 21.5 into your 24-bit floating point type. Your final answer should be given in hexadecimal. Show your work for partial credit!

Miscellaneous

16. [6 points] What is the range of your numerical type from the previous page, both in terms of how many significant digits it can hold, and the (rough) maximum value it hold? Briefly explain why for each. For the latter (maximum value), we just want an estimation. The regular 32-bit floating point type has a maximum value of 3.4×10^{38}, so just saying 10^{38} (or even 10^{37} or 10^{39}) would be sufficient accuracy.

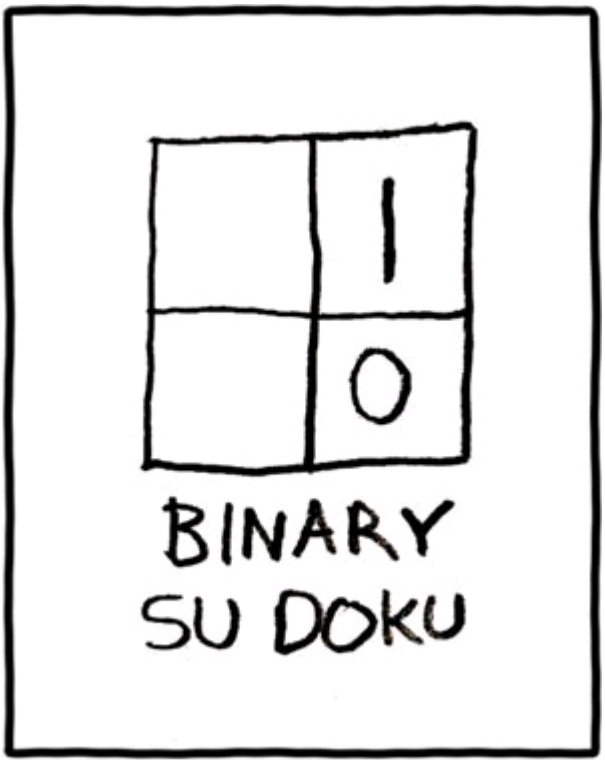
17. [4 points] What is the difference between a *shallow copy* of a data structure and a *deep copy* of a data structure?

18. [4 points] List all the Emacs keyboard commands that you know, and give a BRIEF description of what each one does (a word or two is fine here).

19. [4 points] Why would you want to pass in a `int**` parameter by reference into a subroutine?

This page unintentionally left blank.

But you get two points for bubbling in the form at the bottom of this page. Woo-hoo!



Can you finish the binary Sudoku? It's not required, but it's quite a challenge!

(the bubble footer is automatically inserted in this space)