CS 2150 Final Exam

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, I will rip up your exam and give you a zero.

There are 12 pages to this exam – once the exam starts, please make sure you have all the pages. Pages 2-11 are worth 12 points each, for a total of 120 points. The first and last pages (YOU MUST COMPLETE THEM FOR CREDIT) are worth 7 points, combined. Thus, this exam is worth 127 points – and by the way, that number is prime, and would thus make an excellent hash table size. Note that pages 1, 11, and 12 will take significantly less time than the other pages. With 180 minutes (3 hours) for the exam, and 108 points (not counting pages 1, 11, or 12), you should spend about 1.5 minutes per question point. The long section of reading (page 8 only; not page 6) is worth free points to give you time to read those pages.

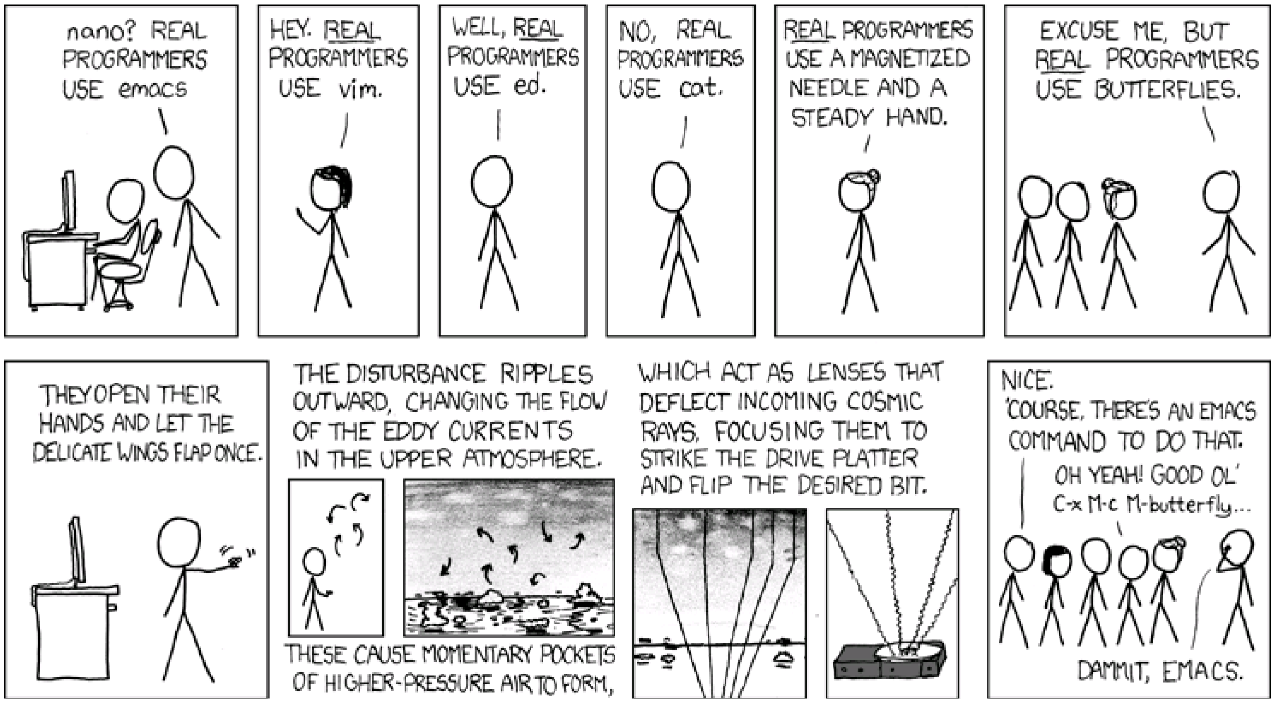
If you do not bubble in a page, you will not receive credit for that page!

This exam is CLOSED text book, closed-notes, **closed-calculator**, closed-cell phone, closed-computer, closed-neighbor, etc. Please sign the honor pledge here:

____

____

____


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

(the bubble footer is automatically inserted in this space)

Numbers

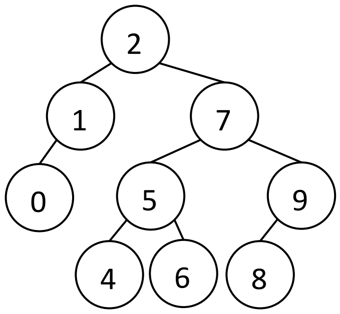
1. [6 points] Convert -84 to a 32-bit **little-endian** two's complement binary representation. Your final answer should be in hexadecimal, though, and not binary. Show your work if you want to receive partial credit!

2. [6 points] Consider the **big-endian** hexadecimal value 0xc1d80000. Interpret this as an IEEE 754 32-bit floating point number, and print the result. Show your work if you want to receive partial credit!

Trees & Hashes

3. [3 points] What is the big-theta running time of a hash table insert with a probing collision resolution strategy?

4. [3 points] Insert 3 into the AVL tree below. Show the resulting data structure.



5. [3 points] Consider the three types of self-balancing trees that we have seen: AVL, red-black, and splay. BRIEFLY give one advantage and one disadvantage of each of these trees. If it's not obvious, explain why.

C++

6. [3 points] What is the difference between shared multiple inheritance and replicated (aka repeated) multiple inheritance?

7. [3 points] Why is dynamic memory allocation so (relatively) slow?

8. [3 points] List all the gdb commands that you know. If you get to 6, you're fine for this question.

9. [3 points] Briefly give a convincing example of when we would want to allow implicit constructor initialization in a C++ class.

---

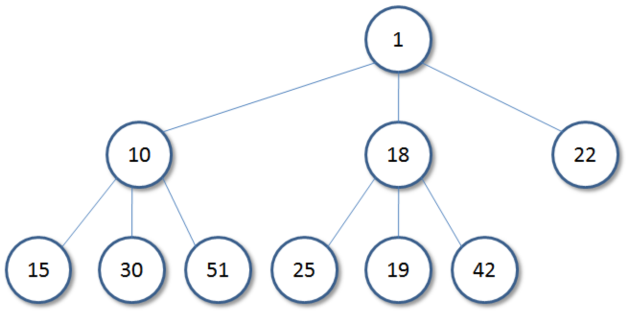
x86

10. [6 points] Draw a complete activation record for a subroutine. For any part that is optional (number of parameters passed, for example), assume that optional part is indeed used. This should be drawn in the same manner as the stack develops (i.e. how we've seen them in lecture).

11. [6 points] The C calling convention on x86 consists of two parts: what the stack should look like (which you answered in the previous question), and certain 'agreements' – the convention – as to how the caller and callee should behave. What are these 'agreements'? If you are confused where to start on this question, walk through the steps of the prologue and epilogue for the caller and callee – the parts that do not affect the activation record's *structure* are these 'agreements'.

Heaps

Consider a new data structure called a *d-ary heap*. A d-ary heap is just like the regular binary heap that we saw in lecture, but each node has (at most) *d* children instead of (at most) 2. Thus, a binary heap is the same thing as a 2-ary heap. A 3-ary heap (also called a ternary heap) is shown in the diagram to the right. A 4-ary heap is such that each node has 0-4 children, etc. All of the other heap operations that we are familiar with (deleteMin(), insert(), etc.) operate the same with a d-ary heap.



As in lecture, heaps can be min-heaps or max-heaps. We will only consider min-heaps, which is what the diagram to the right shows.

Theoretically, *d* can be any positive value. There are advantages and disadvantages to having high *d* values, which is what the following questions focus on.

12. [6 points] Consider the insert() operation on a d-ary heap. How would it be different than with a binary heap? What is the big-theta running time? If the big-theta running time is not accurate indicator of the fact that this operation is faster or slower, briefly explain why.

13. [6 points] Consider the deleteMin() operation on a d-ary heap. How would it be different than with a binary heap? What is the big-theta running time? If the big-theta running time is not accurate indicator of the fact that this operation is faster or slower, briefly explain why.

(the bubble footer is automatically inserted in this space)

Heaps, continued

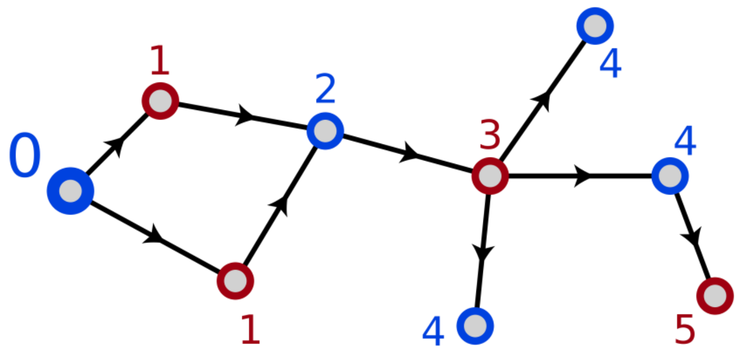
14. [3 points] For an arbitrary value of *d*, how could such a d-ary heap be implemented as an array? Any obvious advantages or disadvantages?

15. [3 points] Consider the case when *d*=4. How could such a d-ary heap be implemented as an array? Any obvious advantages or disadvantages?

16. [3 points] What are an advantage and a disadvantage of having *d* be a high value?

17. [3 points] Considering all of this, when would you want to use a d-ary heap when *d* was a high value?

Graphs



Graph coloring means assigning a 'color' to each graph node. No two adjacent nodes, as represented by having an edge between them, can have the same color. This is used, for example, to draw political maps of countries – each state is a node, and if two states share a common border, then they have an edge between them. By successfully coloring the graph, you can then assign colors to each state (node) so that no two adjacent states (nodes) share the same color.

A *bipartite graph* is a graph that can be colored with only two colors. Consider the image to the right: the graph is colored with two colors: blue and red. This may not render correctly on the grayscale versions of this exam, but all the even numbered nodes are blue and the odd numbered nodes are red. Notice that no two red nodes have an edge between them, and that no two blue nodes have an edge between them.

A triangle is a graph where each point is a node. Such a graph cannot be colored with 2 colors – you color two of the nodes different colors, and there is no color for the third node, since it cannot have an edge to a vertex of the same color. Such a graph is not a bipartite graph.

The numbers on the above graph represent the (shortest) distance from node labeled with zero. This presents one way to label a bipartite graph: perform an single-source shortest path algorithm, and color all odd nodes red and all even nodes blue. We could modify Dijkstra's shortest path algorithm to easily add a color to a node in addition to the shortest distance. We will call this modified algorithm "Dijkstra's Graph Coloring Algorithm" below (although we don't think he would have liked it being called that).

Algorithms that deal with bipartite graphs don't just color them, they need to determine if the graph is bipartite or not. If we added an edge between the two red nodes labeled with a one, we would get the same numbers from the algorithm on each node, but such a graph is not a bipartite graph. This fact will be needed for a few of the questions below. The algorithm we presented above only colors a graph; it does not determine if the graph is bipartite or not.

18. [4 points] What is the running time of Dijkstra's Graph Coloring Algorithm? Why?

Graphs, continued

19. [3 points] How could we modify Dijkstra's Graph Coloring Algorithm to use it to determine if a graph is bipartite or not?

20. [3 points] What is the running time of your algorithm in the previous question? Why?

21. [3 points] Briefly, how would you modify the algorithm you presented above to handle 3-coloring of a graph (i.e. using three colors, instead of two, to color a graph)? This is a hard question to get correct, so don't spend an inordinate amount of time on it!

22. [3 points] Can you come up with a simpler algorithm for determining if a graph is bipartite or not? This is also a hard question, so don't spend an inordinate amount of time on it!

Miscellaneous

23. [3 points] List (and BRIEFLY describe) three *unique* features of Intercal.

24. [3 points] List (and BRIEFLY describe) three *unique* features of the aspect-oriented language AspectJ.

25. [3 points] List 6 doxygen tags, and BRIEFLY explain what they do.

26. [3 points] What are shell scripts good for? What are they not good for? Please give convincing examples here…

Demographics

We meant to ask these in an end-of-the-semester survey, but didn't get to it in time. So we'll put it here for some extra points on the exam!

27. [0 points] Did you put your name at the top of this page? You need to in order to get the points on this page.

28. [2 points] What is your major or minor (if you have not declared, then your intended major or minor)?

- BS CS
- BA CS
- BS CpE
- CS minor
- Neither majoring nor minoring in computing
- Other (please explain): ______________________________

29. [2 points] What CS 1 class did you take? Please circle one.

- CS 101 (now 1110)
- CS 101-E (now 1111)
- CS 101-X (now 1112)
- CS 150 (now 1120)
- AP credit
- Transfer credit
- Placed out of it via the 101/1110 placement exam
- Other (please explain): ______________________________

30. [2 points] If you took your CS 1 class in college (i.e. CS 101/1110, 101-E/1111, 101-X/1112, 150/1120, or a transfer class), in what semester did you take it?

31. [2 points] What CS 2 class did you take? Please circle one.

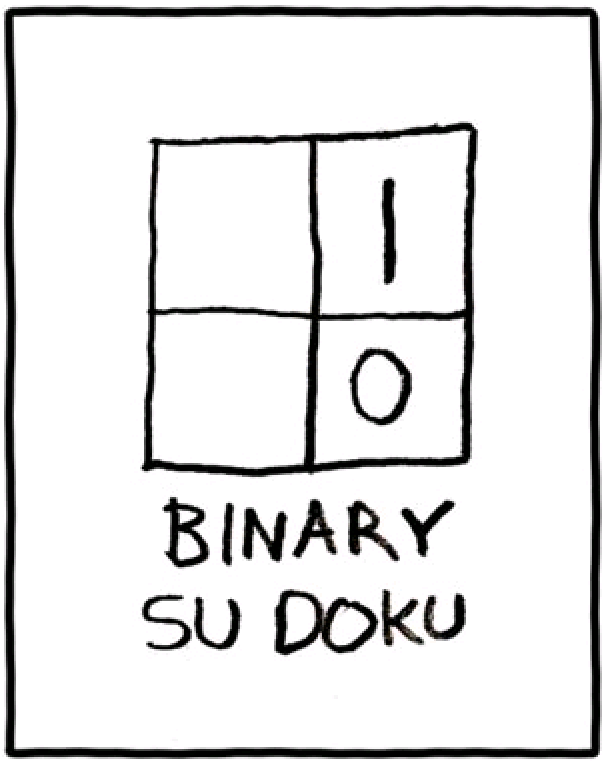
- CS 201 (now 2110)
- CS 205 (now 2220)
- AP credit
- Transfer credit
- Other (please explain): ______________________________

32. [2 points] If you took your CS 2 class in college (i.e. CS 201/2110, 205/2220, or a transfer class), in what semester did you take it?

33. [2 points] Did you attend the review session? You'll get full credit for this question, as long as you answer it honestly (we know *most* of the people that were there, but not all).

(the bubble footer is automatically inserted in this space)

34. [4 points] Can you complete the Binary Sudoku? It's worth 4 points! Insert the digits '0' and '1' into the table such that no row has two of the same digit, and no row has two of the same digit. (This problem is a bit harder when it's a 9x9 table)


1
0
BINARY
SU DOKU