CS 2150 Final Exam

Name ______________________________

You MUST write your e-mail ID on **EACH** page and bubble in your userid at the bottom of this first page. And put your name on the top of this page, too.

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 form. So please do that first. Sorry to have to be strict on this!

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

There are 10 pages to this exam. Once the exam starts, please make sure you have all the pages. Questions are worth different amounts of points.

If you do not bubble in this first page properly, you will not receive credit for the exam!

Answers for the short-answer questions should not exceed about 20 words; if your answer is too long (say, more than 30 words), you will get a zero for that question!

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 below.

____

____

____

____

Serious error.
All shortcuts have disappeared.
Screen. Mind. Both are blank.

(the bubble footer is automatically inserted into this space)

Page 2: Number Representation and Big-OH Analysis

1. [3 points] What is the smallest (positive, non-zero) value of an IEEE 754 *double* precision floating point number? Make sure to show your reasoning here. Leave your answer as an expression in scientific notation.

2. [3 points] Suppose we have a *9-bit (not 8-bit!) two's-complement integer*. What is the largest magnitude positive and negative value that we can represent?

3. [3 points] What is the formal definition of Big-Oh? Give a brief description of the two constants used in this definition.

4. [3 points] What is the formal definition of Big-Theta?

Page 3: C++ and Memory

5. [3 points] Briefly describe the difference between *shared multiple inheritance* and *replicated multiple inheritance*.

6. [3 points] Considering what we learned in the memory slide set (how much faster arrays can be than linked lists in many cases), when, then, are linked lists better than arrays?

7. [3 points] Briefly describe two different examples where a memory leak occurs.

8. [3 points] When discussing caches, we mentioned two properties that caches take advantage off in order to work efficiently. Name and briefly describe them.

Page 4: Exam 2 Stuff

9. [3 points] Do we really need calling conventions for a program to compile? Briefly, what problems might be caused if we did NOT have a calling convention?

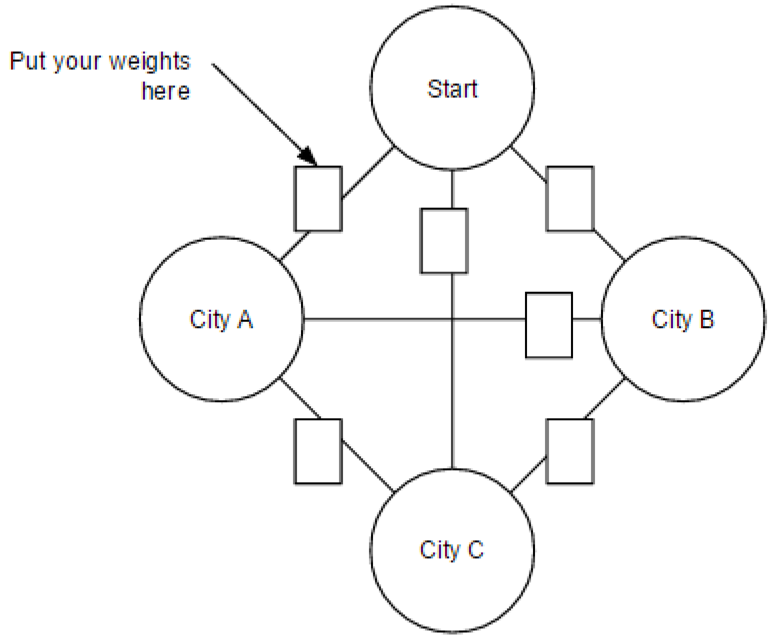
10. [3 points] What is an activation record? Briefly explain why activation records are important for executing functions.

11. [3 points] Recall that *IBCM* provides exactly 4096 memory locations (0-4095). Suppose we double this value to 8192 memory locations, but that EVERYTHING else about the *IBCM* specification remains the same. Name one thing we COULD use these new memory addresses for and one thing we COULD NOT use these locations for. *This is not a trick question. Think about the limits of IBCM given the language specification.*

12. [3 points] List the three requirements of a good *hash function*. For each, briefly describe why NOT having this property would be problematic.

Page 5: Graphs

13. [6 points] Consider the following algorithm that attempts to solve the *Traveling Salesperson Problem*: From the start node, follow the shortest edge to any unvisited city, then repeat. Go back to the start node once all cities are visited. On the following graph, label the edges with weights such that this algorithm would NOT find the best tour. Then, list the tour the algorithm does find AND the tour that is actually the most optimal.



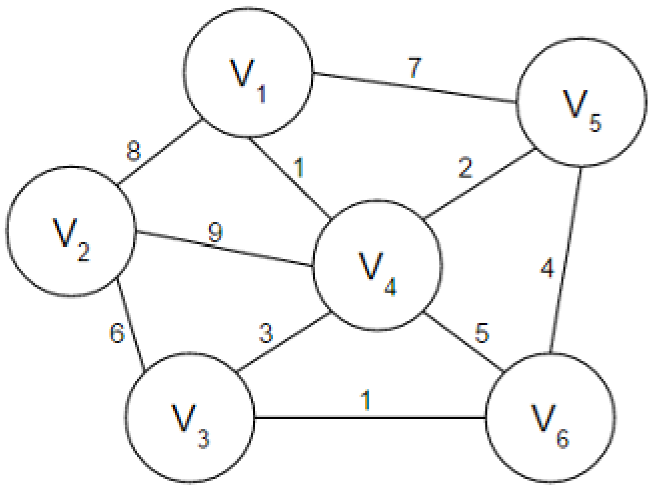
Path Algorithm Returns:	
Actual Best Path:	

14. [3 points] Give the **exact** amount of space used by a graph implemented as an *adjacency matrix* and as an *adjacency list*. Express both in terms of v and e. Clearly state any other assumptions you make.

15. [3 points] State and explain the Big-Theta running time of *topological sort*.

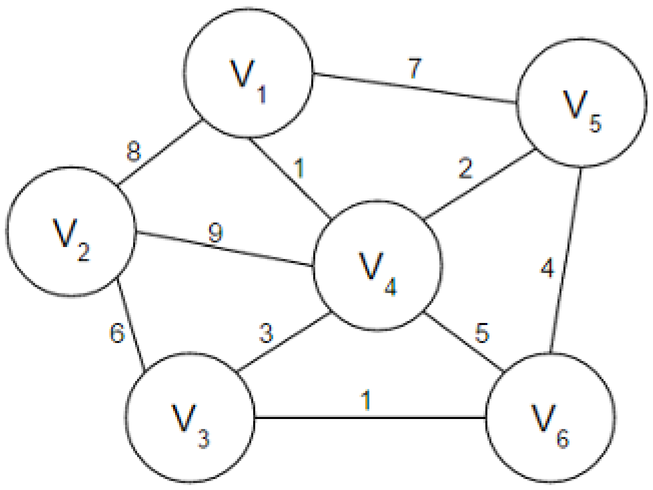
Page 6: More Graphs

16. [6 points] Perform either *Prim's algorithm* or *Kruskal's algorithm* on the following graph to find the *minimum spanning tree*. If you choose *Prim's*, then use V_1 as the start node. Place a number in the boxes below corresponding to the order in which the edges are added to the *MST* as the algorithm executes. If the edge is not part of the *MST*, then leave that box blank.



Edge	Order Added
(V_1, V_2)	
(V_1, V_4)	
(V_1, V_5)	
(V_2, V_3)	
(V_2, V_4)	
(V_3, V_4)	
(V_3, V_6)	
(V_4, V_5)	
(V_4, V_6)	
(V_5, V_6)	

17. [6 points] Perform *Dijkstra's algorithm* on the following graph, given that V_1 is the start node. If any cell in the table contains a value that gets overwritten, show the value crossed out with the new value next to it.



Node	Known	Dist	Path
V_1			
V_2			
V_3			
V_4			
V_5			
V_6			

Page 7: Heaps and Huffman Coding

18. [3 points] Give an example of a non-trivial input string (either by description or by example) that would NOT get very good Huffman coding compression; briefly explain why it would not be good.

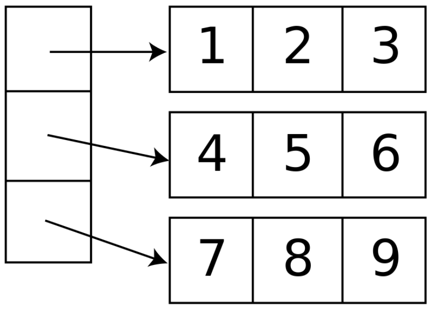
19. [3 points] When percolating down in a *min heap*, why do we swap the given node with the smaller of the two children?

20. [3 points] Suppose we want to add a new method to our *min heap* class called `updatePriority(id, newPriority)`. This method changes the priority of an item that already exists in the heap. What would the run time of this operation be? Make sure to explain your reasoning.

21. [3 points] Pseudo-code a method that sorts a list of integers in $\Theta(n \log n)$ time. You should assume that you have a *MinHeap* class implemented. Your solution MUST use this *MinHeap* class.

Page 8: Iliffe Vectors

There are no questions on this page, but the questions on the next page use this information.

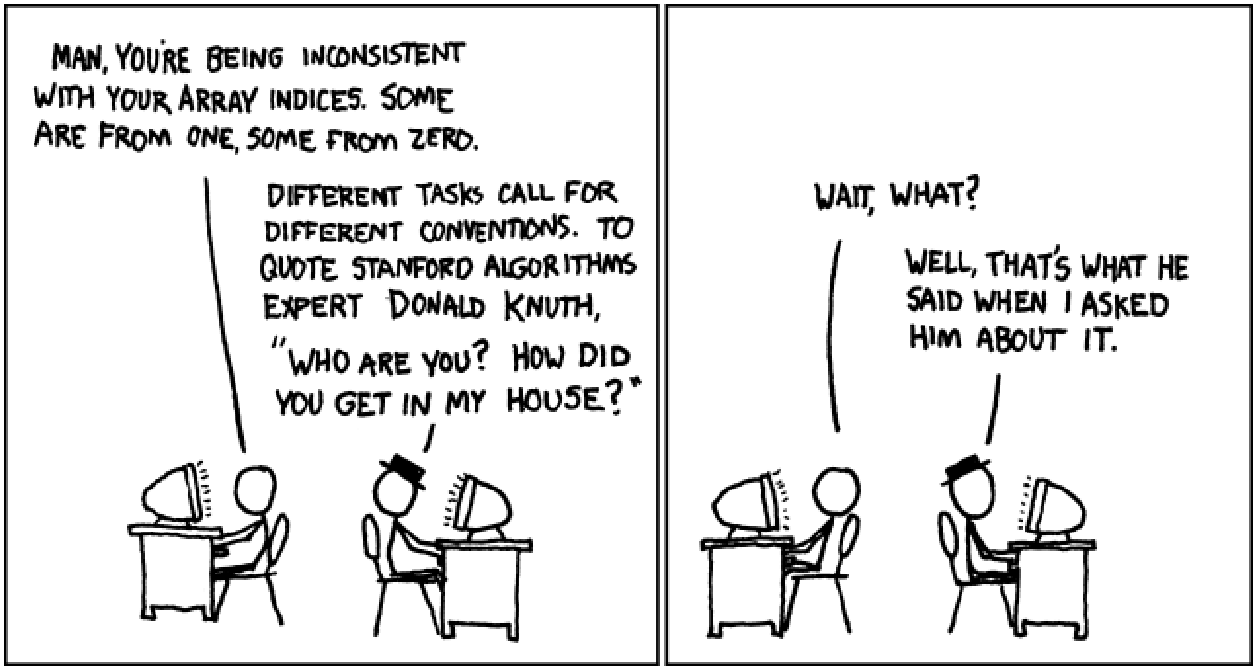


In class, we saw how multi-dimensional arrays in C++ were stored in row-major order in memory. An alternative to this is called an **Iliffe vector**. When allocating a multi-dimensional array, an Iliffe vector allocates one array with pointers to arrays containing the data in the next dimension. For example, a 2-dimensional array with 5 rows and 6 columns would be allocated as an array of 5 pointers, each of which points to a single row array with 6 elements each.

As another example, consider the diagram shown to the right. This is equivalent to a 3x3 array, but is really implemented as a single dimensional array of size 3 of pointers (the vertical one with arrows coming out of it), as well as three single dimensional arrays (the horizontal ones with the numbers) also of size 3.

This may seem like a strange way to store multi-dimensional arrays, but *many* languages do it this way, including: Java, Python (multi-dimensional lists), Ruby, and at least a dozen other popular languages.

Questions about Iliffe vectors are on the next page.


MAN, YOU'RE BEING INCONSISTENT WITH YOUR ARRAY INDICES. SOME ARE FROM ONE, SOME FROM ZERO.
DIFFERENT TASKS CALL FOR DIFFERENT CONVENTIONS. TO QUOTE STANFORD ALGORITHMS EXPERT DONALD KNUTH, "WHO ARE YOU? HOW DID YOU GET IN MY HOUSE?"
WAIT, WHAT?
WELL, THAT'S WHAT HE SAID WHEN I ASKED HIM ABOUT IT.

xkcd # 163

Page 9: Iliffe Vectors

22. [3 points] Give (and briefly describe) a compelling scenario in which using an *Iliffe Vector* would be more efficient than using a multi-dimensional array. You can use either *space* or *time* as the metric for your comparison.

23. [3 points] Fill in the following chart with the run times of the following operations and a short description of why. We assume a two dimensional array of size r by c.

Operation	Run time	Reasoning
find(i,j)		
Allocating a r x c Iliffe vector		

24. [3 points] We know a two dimensional array of size r x c takes up $r * c * s$ bytes, where s is the size of the array cell (we are ignoring the array base name pointer for this question). How much space does an Iliffe vector of size r x c take up?

25. [3 points] Suppose I have a three dimensional *Iliffe Vector* called `A[][][]`. Suppose I want to access element `A[i][j][k]`. Write an expression that evaluates to the proper memory address of element `A[i][j][k]`. If you want to dereference a pointer, use the asterisk (e.g., `*(A[j])`). You may also assume that `A[][][]` stores elements that are 1 byte each.

Page 10: Demographics Name & userid: ______________________________

We meant to ask these in an end-of-the-semester survey, but we did not get to it in time. So we'll put it here for some extra points on the exam! Sorry if this page is a bit crowded. . .

26. [0 points] Did you put your name and userid at the top of this page? You need to do so in order to get the points on this page!

27. [2 points] What is your major or minor? If you have not declared, then answer with your intended major or minor. Please circle one.

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

28. [1 points] Have you already declared the major/minor mentioned above? Circle: Yes or No

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

    - CS 1110
    - CS 1111
    - CS 1112
    - CS 1120
    - AP credit
    - Transfer credit
    - Other (please explain): ________________________
    - Placed out of it via the CS 1110 placement exam

30. [1 points] If you took your CS 1 class in college (i.e. CS 1110, CS 1111, CS 1112, CS 1120, or a transfer class), in what semester did you take it? Please specify a semester by season and calendar year (i.e., "fall 2014" and not "my second year").

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

    - CS 2110
    - CS 2220
    - AP credit
    - Other (please explain): ________________________
    - Transfer credit
    - Placement exam

32. [1 points] If you took your CS 2 class in college (i.e. CS 2110, CS 2220, or a transfer class), in what semester did you take it? Please specify a semester by season and calendar year (i.e., "fall 2014" and not "my second year").

33. [1 points] Did you attend the final exam review session? You'll get full credit for this question, as long as you answer it honestly (we know most that were there, but not all).

34. [2 points] For the 3-credit courses for next semester (not summer or J-term):

    - How many CS courses are you enrolled in (not wait-listed)?
    - How many CS courses are you wait-listed for?
    - How many CS courses would you *like* to be enrolled in?