CS 2150 Final Exam, spring 2016

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.

_____

_____

_____

_____

You step in the stream,
But the water has moved on.
This page is not here.

(the bubble footer is automatically inserted into this space)

Page 2: x86

1. [12 points] Consider the following C/C++ function:

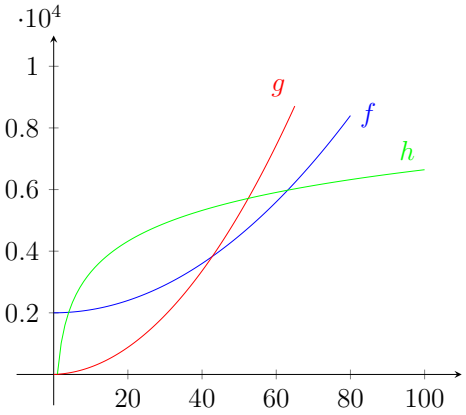
```
int triangle(int x) {
    if(x == 0) return 0;
    else return (x + triangle(x - 1));
}
```

Fill in the x86 assembly code, below. Note that if your response for section `.L2` is longer than 12 opcodes, it will not be accepted (but it can certainly be less than 12 lines). Also note that the `.L3` section wraps to the second column.

```
_triangle:
.LFB0: push    ebp
       mov     ebp, esp
       cmp     [ebp+8], 0

       jne     __________

       mov     eax, __________

       jmp     __________

.L2:   ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________

       ____________________
```

```
.L3:
       leave
       ret
```

Page 3: Asymptotics



2. [3 points] Consider the plot above. Determine whether the following relationships are true or false by circling the right answer. Note that the two bottom ones are little-oh and little-omega. Also note that the plots do not cross beyond what is shown in the graph above.

$f \in \mathcal{O}(g)$	:	true	false	$f \in \Omega(g)$	:	true	false
$h \in \mathcal{O}(f)$	:	true	false	$g \in \Omega(f)$	:	true	false
$g \in o(h)$	:	true	false	$f \in \omega(h)$	:	true	false

3. [3 points] Let $f(n) = 2 \cdot n + 1$ and $g(n) = 2^n$. **Prove** that $f \in \mathcal{O}(g)$.

4. [3 points] Briefly explain the relationship between $\mathcal{O}()$ and $o()$ as well as $\Omega()$ and $\omega()$. Why does defining a "little-θ" version of $\Theta()$ in an analogous way not make sense?

5. [3 points] Give a compelling instance of when you might use big-Omega in practice.

Page 4: C++

6. [5 points] The following code segment contains an error. Briefly describe what is the error in this method, and briefly describe how can the code be changed to fix this. Note that the error is *not* a compilation error nor a run-time error.

```
class TreeNode {
public:
    TreeNode *left;
    TreeNode *right;
    int val;
};
// for the following function, the parent of
// what is returned is set by the caller
TreeNode *rotateLeft(TreeNode *p) {
    TreeNode *rchild = p->right;
    p->right = NULL;
    rchild->left = p;
    return rchild;
}
```

7. [5 points] The following code segment contains an error. Briefly describe what is the error in this method, and briefly describe how can the code be changed to fix this. Note that the error is *not* a compilation error nor a run-time error.

```
bool present(const string &s, string **table, int tableSize) {
    unsigned int h = hash(s) \% tableSize;
    while (table[h] != NULL) {
        if (s == *table[h])
            return true;
        else
            return false;
        h = (h+1) % tableSize;
    }
    return false;
}
```

8. [2 points] List a run-time error *other* than segmentation fault, and briefly describe what can cause it

Page 5: Trees, Advanced C++

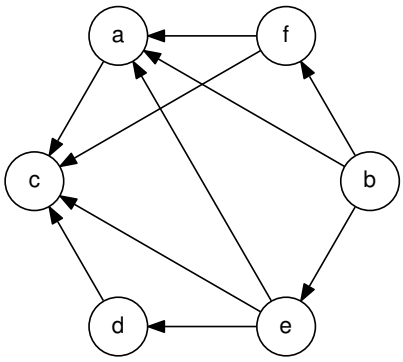
9. [6 points] Write `findSumPaths(BTNode* node, int targetSum)`, a *recursive* C++ function, that is given a binary tree node (not a BST) and a given sum. Assume that `BTNode` objects contain an integer value `val`, and `left` and `right` child pointers. This method should print out every path in the tree from the root to any leaf node whose sum is exactly the given parameter `targetSum`. Note that there may very well be more than one path, and you should print them all out in any order. You may also add additional parameters to the method header if you'd like, to make your recursion work better.

10. [3 points] List one advantage and one disadvantage for both static and dynamic dispatch. Note that you can't use the same reason twice! So if *A* is faster than *B*, you can't *also* say that *B* is slower than *A*.

	Advantage	Disadvantage
static		
dynamic		

11. [3 points] In class, we saw three (at least) different uses for the `virtual` keyword in C++. List them.

Page 6: Graphs



12. [4 points] Write the adjacency matrix for G, shown above, and the indegrees for every node.

13. [4 points] As part of topological sort, it is important to compute the indegrees of each node in the graph. Given the following function signature, write valid C++ code to compute the indegree of each node in the graph and store it in the `indegree` array. Assume `adjMat` is an adjacency matrix already initialized in another function, `n` is the number of vertices in the graph, and `indegree` is a zero-filled array that your function is to modify. The reference solution is 3 lines long; aim for conciseness as well as correctness.

```
void computeIndegrees(int[][] adjMat, int n, int[] indegree); {
```

14. [4 points] Give a topological ordering for the graph G shown above. Is there always only one topological ordering?

Page 7: Heaps & Huffman Coding

15. [3 points] Decode the Huffman encoded string "10110000011111101110" with the following dictionary: f: 0, a: 100, h: 101, u: 110, m: 1110, n: 1111.

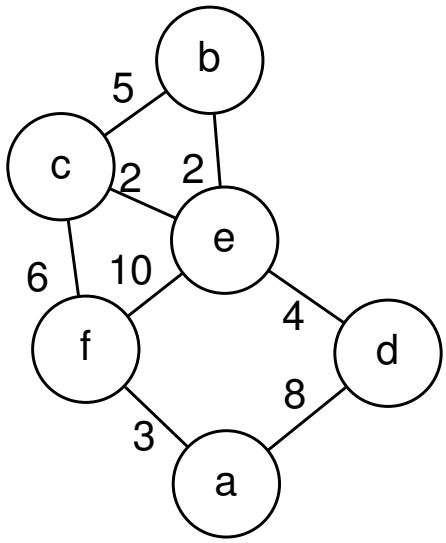
16. [3 points] When is it possible to get a totally unbalanced Huffman encoding tree? Be brief! You can provide an example, if desired.

17. [6 points] Assume that you have a `minHeap` class that has the standard three priority queue operations (insert, remove, find). This class stores only `int` values, which are also the priority. Write a heap sort, in a `vector<int> heapSort(vector<int> v)` function, that uses this class. Note that the `MinHeap` class does not have any constructors other than the default constructor. This method should be short – it can be written in 4 lines (but yours doesn't have to be that short).

Page 8: Graphs & Miscellaneous

18. [6 points] Given the following graph, perform Dijkstra's shortest path in the table below. If a cell is updated, be sure to include both the original value(s) (crossed out), as well as the updated value(s). Start at node "a".

node	known?	distance	path
a		0	
b			
c			
d			
e			
f			



19. [3 points] List 8 debugger commands, and *very* briefly describe what they do. You can use gdb or lldb commands, but you have to list all 8 commands from one or the other – you can't mix and match.

20. [3 points] Briefly, why did we make you use a UNIX environment (Linux, Mac OS X command line, etc.) in this course? Briefly, why did we have you avoid IDEs?

Page 9: No Questions Here

This page unintentionally left blank

Feel free to use the page for scratch paper; however, answers on this page will not be graded.

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...

21. [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!

22. [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

23. [1 points] Have you already declared the major/minor mentioned above? Circle: Yes or No

24. [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

25. [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").

26. [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

27. [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").

28. [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).

29. [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?