CS 2150 Final Exam, spring 2017

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.

_____

_____

_____

_____

In theory, there is no difference between theory and practice.
But, in practice, there is.

(the bubble footer is automatically inserted into this space)

Page 2: C++

1. [3 points] Suppose vectors worked slightly differently than defined in the course. Instead of doubling in size every time the vector fills up, they instead grow by ten in size. Is the amortized cost of inserting into this vector still constant? Why or why not?

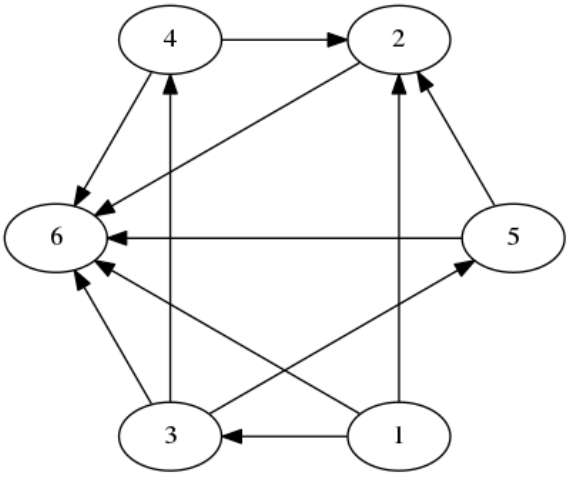
2. [3 points] What are the three pointer dereferences needed for dynamic dispatch?

3. [6 points] We have discussed two categories that are used to define variables: lifetimes of scoped vs. unlimited, and the compile-time size of known vs. unknown. Write a *small* C/C++ program that has one variable of each possible combination. This should be a **SHORT** program (if it's more than 10 lines we won't grade it). To make it easier to grade, please only have one function, `main()`, and don't worry about `namespace` or `#include` lines. Please have the variables names be the following (the types are your choice):

   - variable `a`'s lifetime is scoped and compile-time size is known
   - variable `b`'s lifetime is scoped and compile-time size is unknown
   - variable `c`'s lifetime is unlimited and compile-time size is known
   - variable `d`'s lifetime is unlimited and compile-time size is unknown

Page 3: Graphs and Memory

4. [6 points] Give a topological ordering for the graph below. Is there only one topological ordering?



5. [3 points] Give a scenario where an adjacency list representation of a graph is more space efficient than an adjacency matrix representation.

6. [3 points] With regards to caches, what are *spatial* and *temporal* locality? Why do they matter?

Page 4: Heaps and Huffman coding

7. [3 points] Why is the insert into a binary heap / priority queue expected to be constant time?

8. [3 points] What is the difference between a binary heap and a priority queue?

9. [3 points] Consider determining the Huffman encoding for the input "abdsabbbadq". Draw the initial min-heap used in the algorithm (and indicate the frequencies therein).

10. [3 points] Again consider determining the Huffman encoding for the same input as above: "abdsabbbadq". Draw the final Huffman coding tree determined by the algorithm.

Page 5: Assembly and Memory

11. [8 points] Given the C++ function below, fill in the missing lines for the corresponding x86 implementation.

```
int fib(int n) {
  if (n == 0) return 0;
  if (n == 1) return 1;
  return fib(n-1) + fib(n-2);
}
```

```
fib:
  push rbx
  xor rax, rax
  cmp rdi, 0
  je zero
  ---------
  je one
  ---------
  sub rdi, 1
  call fib
  ---------
  pop rdi
  ---------
  call fib
  add rax, rbx
  jmp done
zero:
  mov rax, 0
  jmp done
one:
  mov rax, 1
done:
  ---------
  ret
```

12. [4 points] What is a memory leak? Why is it a problem? What causes one? Where (in terms of parts of memory) does it occur? List the four answers, but keep them BRIEF.

Page 6: Disjoint sets

This page contains several problems related to the "Disjoint Set" data structure. Read this page carefully, then proceed to the next page to answer questions about it.

In certain algorithms, we are confronted with the problem of performing multiple unions of "disjoint sets" (i.e. sets that contain no common elements) on N items. There are two operations we care about:

1. `union(a, b)` : Join the two sets containing `a` and `b`.
2. `check(a, b)` : Returns `true` if (and only if) `a` and `b` are in the same set.

We will limit this problem to sets containing contiguous, non-negative integers starting from 0. For example, suppose we have $N = 10$; we start with the numbers 0 through 9, each in their own sets:

$$\{0\}\ \{1\}\ \{2\}\ \{3\}\ \{4\}\ \{5\}\ \{6\}\ \{7\}\ \{8\}\ \{9\}$$

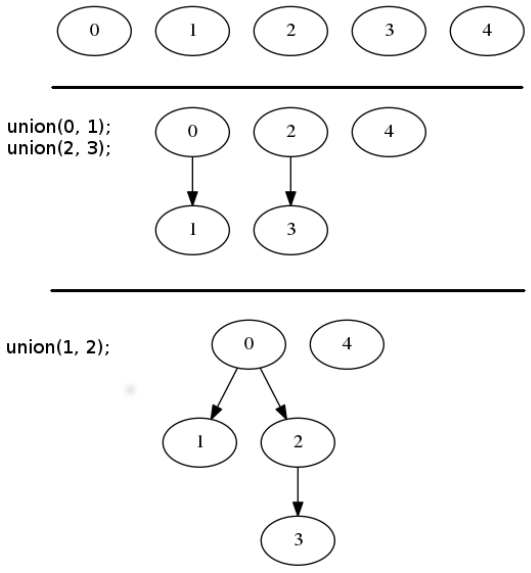
Next, we perform four union operations: `union(0, 1)`, `union(2, 3)`, `union(4, 9)`, `union(1, 3)`. At this point, the sets are now:

$$\{0,1,2,3\}\ \{4,9\}\ \{5\}\ \{6\}\ \{7\}\ \{8\}$$

Note that the last call, `union(1 3)`, performed the union of the sets that contained 1 and 3 (namely, the sets $\{0,1\}$ and $\{2,3\}$).

Finally, we may perform any number of calls to `check` on the sets just shown:

```
check(0, 3) = true
check(3, 4) = false
```



One approach uses a collection of trees (called a *forest*) to represent the sets; each tree in the forest represents a single set. To start, we create N trees, each with only one node. When we call `union(a, b)` we find the roots of the trees containing `a` and `b`, then make the root of the shorter tree a child of the root of the taller tree. If the trees are the same height, then make the root of `a` the parent of the root of `b`. (See the image at right for an example with $N = 5$ during and after a series of operations.)

To do `check(a, b)`, we see whether `a` and `b` have the same root. In the example in the image at right, if we call `check(1, 4)`, we find the root of `1` is `0`, and the root of `4` is `4`, so they are part of different sets. However, a call to `check(1, 3)` finds the root of `1` is `0` and the root of `3` is `0`, which means they are in the same set.

For a memory-efficient data structure, we represent this tree as a *parent array*. If `parent` is our array, then `parent[k]` contains the direct parent of the node k in the tree. The parent of a the root of a tree is, by convention, itself. So, the parent array the example above is:

```
parent = [0, 0, 0, 2, 4]
```

Page 7: Disjoint Sets

13. [3 points] Consider a Disjoint Set data structure with $N = 10$, starting with all items in separate sets. We perform the following operations in sequence: `union(0, 4),union(3, 6), union(9, 7),union(2, 1),union(4, 2),union(6, 0)`. Draw the tree representing the final state of the set and write the parent array associated with this tree.

14. [3 points] What is the big-Theta run time of `union` and `check` (in terms of N) and why?

15. [3 points] One might argue that a linked list would be easier to use than these trees. Why do we use trees instead?

16. [3 points] Write C++ code for `check(a, b)`. Assume there is a global array `parent`, properly filled with values, available. The prototype is: `bool check(int a, int b)`

Page 8: Data structure comparisons

17. [8 points] Fill in the grid with letters describing the advantages and disadvantages of each data structure from the options below. Not every letter has to be used, but you can not use the same letter more than once.

	Advantage	Disadvantage
BST		
AVL Tree		
Linked List		
Hash Table		
Splay Tree		

Options:

A Requires an ordering relation on the data

B Linear-time "contains" operation

C Quadratic-time insertion

D Requires n^2 memory in proportion to number of items

E Quadratic-time "contains" operation for least-recently-accessed entries

F Logarithmic insertion and find

G Guaranteed constant time insertion

H In practice, nearly constant-time "contains" operation

I Worst case linear-time insertion

J Quick retrieval of recently accessed data

K Insertion requires "splitting" data into many pieces

L Performance depends significantly on ordering of input data

M In practice, performs slightly better than linear time for "contains" operations.

N Performance depends significantly on ordering of calls to "contains"

18. [4 points] What is the best collision resolution strategy? Why?

Page 9: 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...

19. [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!

20. [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

21. [1 points] Have you already declared the major/minor mentioned above? Circle: Yes or No

22. [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

23. [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").

24. [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

25. [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").

26. [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).

27. [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?

Page 10: No questions here

This page unintentionally left blank.

xkcd #683