CS 2150 Exam 2, 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 6 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: Exam 1 Material

1. [3 points] How, if at all, would the *implementations* of `void List::printList()` and `void printList(List& source)` differ? If they would differ, then please explain why this difference is necessary.

2. [3 points] Give a compelling situation where you would want to pass a pointer into a subroutine by reference.

3. [3 points] Other than syntax, list the three primary differences between pointers and references.

4. [3 points] What is the min and max value of a 9-bit signed twos-complement integer?

Page 3: Trees

5. [3 points] Describe three situations when trees are *NOT* a good data structure to use?

6. [3 points] Splay trees are $\Theta(\log n)$ *amortized* time. What does *amortized* mean here?

7. [6 points] For each of the tree types below, give one advantage and one disadvantage. 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
BST		
AVL		
Red-black		
Splay		

Page 4: Hashes

8. [3 points] Write the English description of an algorithm that will find the k^{th} highest value in a hash table. We don't want C++ code here! We assume that the keys and values are the same (like in lab 6), and that they can be compared with a simple $<$ (like ints or strings).

9. [3 points] What is thrashing? Use an example of a hash function and table size to demonstrate how this might occur. Use a small hash table so that it will fit below (size $<= 6$).

10. [6 points] In this problem, you are asked to simulate a hash table that stores `Person` objects using double hashing. The hash table size is a fixed value of 10. Write the name of each inserted person in the appropriate cell.

    For a primary hash function (h_1), use the last two digits of the birth year and the number of the birth month as the secondary hash function (h_2). For example, suppose David was born on June 22, 1932 (6/22/1932). Then, $h_1(\text{David}) = 32$ and $h_2(\text{David}) = 6$.

    Insert the following values (in order) into the provided table (birth dates in MM/DD/YYYY format following the name):

    1. Andrew (05/13/1999)
    2. Aaron (01/17/1903)
    3. Mary (08/23/1939)
    4. Becky (12/06/2017)
    5. Eve (04/18/1990)

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

Page 5: Assembly

11. [3 points] Consider the x86 `push` command. Implement this command using more basic opcodes (i.e., implement this command using opcodes *other than* `push`). For the sake of this question, we are pushing the variable "`[var]`", which is a 8-byte variable (a C++ `long` or `double`, for example).

12. [6 points] Vecsum has returned, but this time, its recursive! The x86 code on the right should have the same functionality as the C++ function on the left. Fill in the missing instructions.

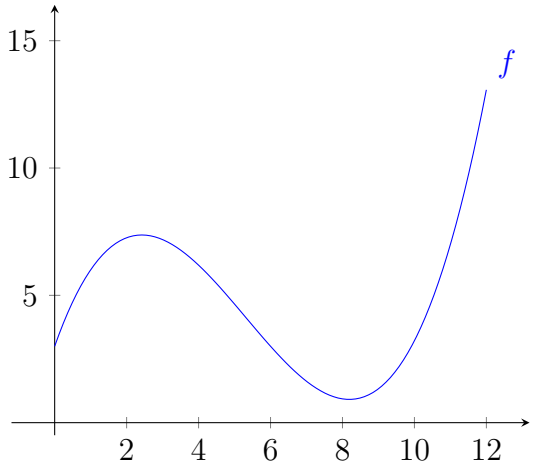
```
int vecsum (int arr[],
            int n,
            int i) {
  if (n == i)
    return 0;
  return arr[i] +
      vecsum(arr,n,i+1);
}
```

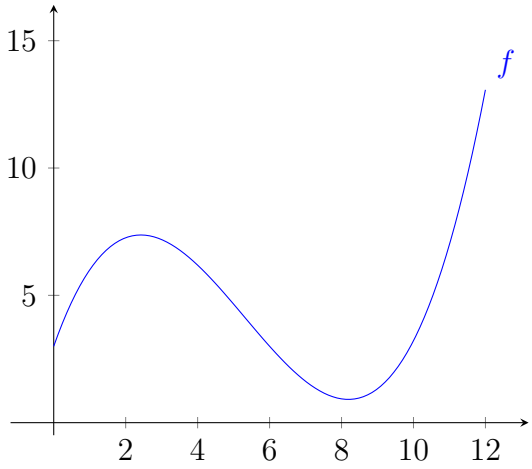
```
vecsum:
    xor rax, rax

    -------------
    je done

    -------------
    mov rbx, [rdi+8*rdx]

    inc ____
    call vecsum

    -------------
    pop rbx
done:
    ret
```

13. [3 points] Briefly describe how a buffer overflow attack works at the assembly level.

Page 6: Miscellaneous

14. [4 points] The following plots depict the graph of a function f. On the **left** plot, graph a function g such that $g \in O(f)$. On the **right** plot, graph a function h such that $h \in \Omega(f)$. You can assume that f continues to increase (at roughly the same rate) as it goes beyond the edges of the graph shown.





15. [4 points] IBCM doesn't have a way to load a non-constant address without using *self-modifying code*, as was done in the lab. Suppose we added a new instruction LOADI (load indirect) to IBCM that interprets the number in the accumulator as an address, looks up the memory at that address, and replaces the accumulator with the value stored at that address. Using this new instruction, write a snippet of IBCM code (in human-readable opcodes, no numbers) that loads the i^{th} value from an array at base address *a* into the accumulator. Assume *i* and *a* are pre-defined labels.

16. [4 points] Give three cases where big-Theta analysis breaks down.