# CS 2150 Final Exam, spring 2022

## Name _____

You MUST *clearly* write your name above. You must also write your e-mail ID on **EACH** page.

If you are still writing when "pens down" is called, your exam will not be graded – even if you are still writing to fill your name and userid. So please do that first. Sorry to have to be strict on this!

There are 8 pages to this exam. Once the exam starts, please make sure you have all the pages. Questions are worth different amounts of points.

**Answers for the short-answer questions should not exceed about 20 words; if your answer is too long (say, more than about 30 words), you will get a zero for that question! This is not a hard maximum, but you should try your best to answer the question as concisely as you can.**

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.*

## Page 2: Exam 1 material

1. [3 points] *Briefly,* why do we return 0 at the end of the `main()` function?

2. [3 points] *Briefly,* why will a dangling pointer likely *not* cause a segmentation fault?

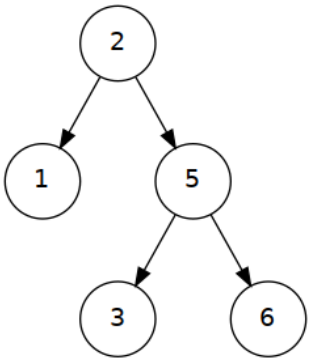
3. [3 points] *Briefly,* when is the copy constructor called versus when is `operator=()` called?

4. [3 points] *Briefly,* when is a linked list preferable to use over a vector?

## Page 3: Exam 2 material

5. [6 points] Convert -26.5 into a 32-bit IEEE floating-point number. Your answer should be in (big-Endian) hexadecimal.

6. [3 points] Insert 4 into the AVL tree shown below. Show the resulting tree.




7. [3 points] Consider the assembly code snippet below. *Briefly* list three errors in the code. By error, we mean anything that will crash the program, is incorrect syntax, doesn't follow the calling convention, etc.

```
mystery:
        mov r10, rsi
        pop rsp
        mov rbx, rdi
        xor rax, r10
        lea rax, [rax + rdi * 6]
        ret
```

## Page 4: Heaps & Huffman coding

8. [6 points] Construct a Huffman Tree for the phrase "to be or not" (without the quotes). Note that spaces are included in the encoding. Afterwards, fill out the table shown. Note that "total bits" column is the total number of bits that *all* instances of that character.

| Character | Frequency | Bit Encoding | Total bits |
|---|---|---|---|
| t | | | |
| o | | | |
| space | | | |
| b | | | |
| e | | | |
| r | | | |
| n | | | |

9. [3 points] *Briefly,* why is the expected runtime for insert into a binary heap constant?

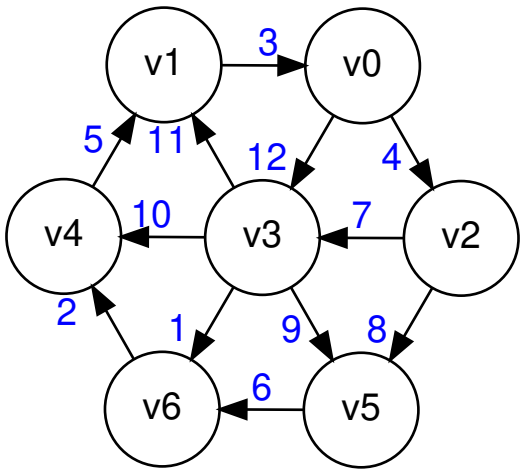
10. [3 points] *Briefly,* what are the differences between a binary heap and a binary search tree?

## Page 5: Graphs

11. [3 points] Why doesn't Dijkstra's work with negative weight edges? Keep your answer brief! You can answer with a description or you may provide a small graph as a counter-example.

12. [3 points] *Briefly* explain how Kruskal's algorithm to find the Minimum Spanning Tree (MST) works. Hint: the MST algorithm that is much like Dijkstra's shortest path algorithm is Prim's, not Kruskal's.

13. [6 points] Perform *Dijkstra's algorithm* on the following graph, given that $V_0$ 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 |
|---|---|---|---|
| $V_0$ | | 0 | |
| $V_1$ | | | |
| $V_2$ | | | |
| $V_3$ | | | |
| $V_4$ | | | |
| $V_5$ | | | |
| $V_6$ | | | |

## Page 6: Miscellaneous

14. [3 points] *Briefly,* list three situations we encountered this semester where big-Theta does not work well.

15. [3 points] *Briefly* explain the difference between spatial locality and temporal locality when it comes to caches.

16. [3 points] *Briefly,* what are all the possible contents of an activation record in the C calling convention? We have five sample answers in mind, some of which are optional (not necessarily put on the stack) depending on the specific function's needs.

17. [3 points] *Briefly,* list one advantage and one disadvantage to having more cache memory in a computer.

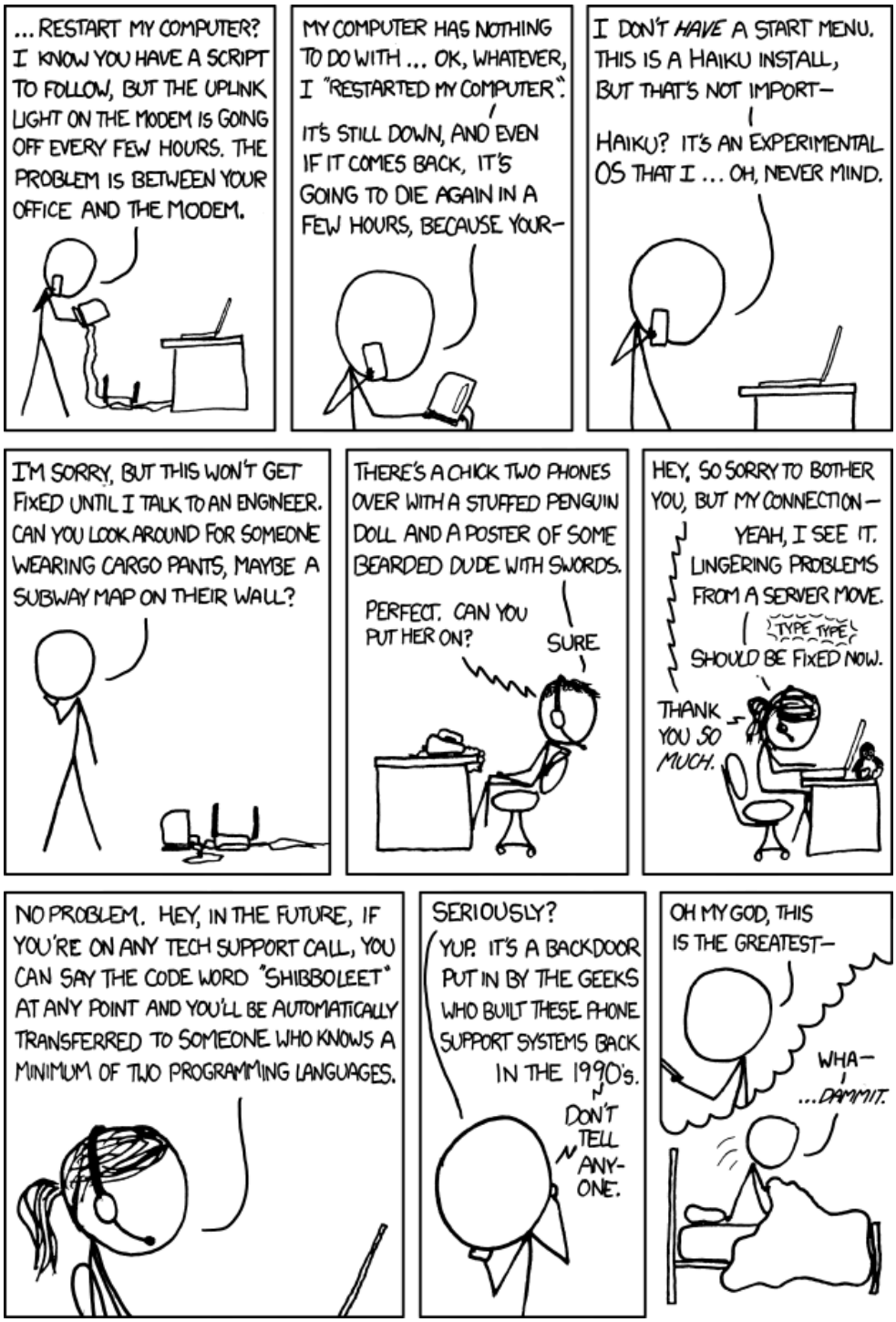
## Page 7: OMG Table

18. [12 points] For *each* data structure we have studied, fill in the worst-case (or, if appropriate, amortized) running times for the three primary operations for the listed data structures. Note that they should be interpreted appropriately for each data structure, so insert/remove/find for a stack are `push()`/`pop()`/`top()`, find for a queue is `enqueue()`/`dequeue()`/`front()`, etc. Write the running time as just $n$, not $\Theta(n)$ and not "linear" (and appropriately different if it's not linear). Any remove function has an iterator, if one is appropriate for that data structure. Any insert is considered to be at the most efficient "end" of the data structure. Lastly, if you are listing an amotized runtime for a given operation, be sure to put the word 'amortized' the box).

| Data structure | Insert | Find | Remove |
|---|---|---|---|
| Array | | | |
| Vector | | | |
| Singly linked list | | | |
| Doubly linked list | | | |
| Stack, vector-based | | | |
| Stack, linked-list based | | | |
| Queue, vector-based | | | |
| Queue, linked-list based | | | |
| Binary search tree | | | |
| AVL tree | | | |
| Red-black tree | | | |
| Splay tree | | | |
| Hash table | | | |
| Binary heap | | | |

## Page 8: Nothing to see here

This page unintentionally left blank.




XKCD # 806