CSC190: Exam 2018-04 (150 minutes)

First name (please write as legibly as possible within the boxes)

Last name

Student ID number

Instructions

- For full credit, you must show all steps and present solutions clearly.
- Solutions must employ the concepts and methods presented in the lectures and labs of this course.
- You MUST use pen. Pencil is NOT permitted on this exam. Don't detach the pages.
- Attempt and answer ALL 18 questions on this 20 page exam.
- Closed book; no aids; no electronic equipment allowed (cellphones, tablets, computers, calculators, etc.).
- The exam is self-contained: absolutely no questions will be answered by the examiner.
- LEGIBLY write your official first name, last name, student number using UPPERCASE letters and numbers.

Possibly useful information about Python

```
Types: sequential (list, string), numeric (integer, float),
       boolean (True, False)
Operators: tests (==, <>, <=, >=, >, <), and, or, not,
           +, -, *, /, %, assignment (=), list copy (L=list(M))
Extending the language: functions (def ... return), classes
```

Possibly useful information about C

```
Types: numeric (int, float), char, struct,
       arrays (TYPE NAME[SIZE]), pointer/address (TYPE *NAME),
Operators: tests (==, !=, <=, >=, >, <), &&, ||, !, +, -, *, /, %,
           assignment (=)
Extending the language: functions structs

Type Analysis:
  TYPE *X; means: *X is a TYPE
                   X is a TYPE *
                  &X is a TYPE **

Passing pointers:
 if the function is defined: RVAL_TYPE FNAME(TYPE1 *PARG1_NAME,...);
 if the caller defines:
    TYPE1 X;
    RVAL_TYPE R;
 then the caller must call FNAME via:
    R = FNAME( &X, ...)

 Since PARG1_NAME is a TYPE1 *, then inside of FNAME you can write:
     *PARG1_NAME = <some value>;
 and the value of X in the caller (which is outside of FNAME) will be set
 to that value.

 If you want to return a value of type TYPE via the arg list,
 then you must define the argument for that value as a TYPE *

Malloc syntax: if you define TYPE *X;
 then you can allocate memory for X via:
   X = (TYPE *) malloc(sizeof(TYPE) * <SIZE>);

 Then X[0] refers to a value of type TYPE at the start of that memory.
      X[<SIZE>-1] refers to the final value of type TYPE in that memory

sizeof: char (1 Byte=8 bits), int/float (4 Bytes), pointer (8 Bytes)

((int)x) provides the integer value of x (e.g., if x is a char)
```

1. "Engineering is a discipline that combines technical skill, mathematical and scientific knowledge, and lucid, clearly-outlined, unemotional, objective, rational thought processes to yield robust solutions to novel problems." Is this True or False? If False, explain why. Otherwise, you may use this page for draft work (your final solutions must appear on the correct page, however).

2. This question concerns stacks and queues. Consider the following operations: $\rightarrow L$, $\rightarrow R$, $\rightarrow A$, $\leftarrow$, $\rightarrow Q$, $\rightarrow S$, $\leftarrow$, $\leftarrow$, $\rightarrow Z$, $\leftarrow$, $\leftarrow$

   - **5 points** Let $\rightarrow$ represent the **enqueue** operator, and $\leftarrow$ represent the **dequeue** operator (which returns the relevant element of the data structure). What is the last return value above? Your work must be shown.

   - **5 points** Let $\rightarrow$ represent the **push** operator, and $\leftarrow$ represent the pop operator (which returns the relevant element of the data structure). What is the last return value above? Your work must be shown.

3. **(10 points)** Complete **all** empty cells of the following table. The values in each column are displayed in the base indicated by the column heading.

decimal (base-10)	hexadecimal (base-16)	binary (base-2)
892		
	0x1af	
		1000010111
	0x39b	
353		

4. **(20 points)** Write a C function, `blend`, that will take in two unsigned ints (`a` and `b`), and return an unsigned int such that bit i of the result is equal to $f(i, x, y)$, where x is bit i of a and, y is bit $31 - i$ of b and f is defined as:

   - (when i is one of $0 + 4j$): x AND y
   - (when i is one of $1 + 4j$): x OR y
   - (when i is one of $2 + 4j$): x XOR y
   - (when i is one of $3 + 4j$): x

   for j in $[0, 7]$. For full marks, you must use the subset of C we covered in class to write an iterative algorithm that increments with a step size of 1 to generate the result.

5. **30 points** Write a Python class for graphs that uses an adjacency matrix representation which must include the following class functions:

   - `__init__` (arg: n; this will create an adjacency matrix for a graph with n vertices)
   - `addEdge` (args: fromVertex, toVertex, weight; this function will create an edge directed from fromVertex to toVertex, with the indicated positive weight)
   - `isEdge` (args: fromVertex, toVertex; this function queries the graph as to whether an edge directed from fromVertex to toVertex exists in the graph)
   - `shortestPath` (args: fromVertex, toVertex; this function will return the shortest path from fromVertex to toVertex)

   You must write this class from scratch, using the primitive Python variable types (lists, integers) and control/datapath facilities. You may only use the subset of Python discussed in class and you may not use existing Python functions with a '.' in the name (e.g., .append). You can assume the existence of a function called `reverse` which will return the reverse of a list argument. You can use the standard Python operators for indexing and concatenation. You may assume that a function called `dijkstra` has already been written for this class; this function takes a single argument (the starting vertex), and returns a 2-list where the 0th component is the list of shortest path lengths, and the 1st component is the prior vertex (where these are defined exactly as we did in class). The vertices of the graph are represented as non-negative integers, starting from 0; weights are assumed to be positive (a zero weight means no edge exists). All functions, except for `shortestPath`, will return either True or False (True indicates the truth of a query or the absence of an error, as the context suggests; False indicates the falseness of a query or the presence of an error).

Answer Python graph class question here:

6. **(2 points)** In the bitwise operation lab, you were shown how simple encryption is accomplished. There was a binary bitwise operator, @, such that $(data@password)@password = data$. What is @?

7. Consider the C function: `float f(int x,float y,char z);` and using C syntax, answer the following:

   a) **2 points** What is the signature of this function?

   b) **2 points** How would you define (not assign) a variable that was a pointer to this function?

   c) **2 points** How would you assign the pointer variable you defined above, to point to `f`?

   d) **2 points** How would you call the function via the pointer you assigned above, with arguments 10,100.0,'x'?

8. For the following questions on time complexity, you must:

   - circle the big-O time complexity that you are reporting as your answer
   - indicate your reasoning

   Where needed, assume that all trees, lists, etc., have n elements.

   - **(5 points)** F is a function that takes a binary tree as input, where the value at each node is an integer. F traverses the tree, and returns the sum of the values found at each node in the traversal. What is the big-O time complexity of this function?

   - **(5 points)** F is a function that takes a binary tree as input, where the value at each node is an integer. F traverses the tree, and returns the sum of the even values found at each node in the traversal. What is the big-O time complexity of this function?

   - **(5 points)** F is a function that takes a sorted list (whose sort direction is known) as input and returns the value of the smallest element in the list. What is the big-O time complexity of this function?

- **(5 points)** F is a function that takes a binary tree as input, where the value at each node of the tree is an list (unsorted) of integers. F traverses the tree, and returns the sum of the smallest element in each list visited in the traversal. What is the big-O time complexity of this function?

- **(5 points)** F is a function that takes a binary tree as input, where the value at each node is a list of integers. F returns the tree of lists, but with all of the lists sorted. What is the fastest algorithm we discussed in class to do a sort, and using this algorithm, what is the big-O time complexity of F?

- **(5 points)** For the prior question: what is the slowest sorting algo we discussed in this course, and using that what is the big-O time complexity of F?

- **(6 points)** What is the time complexity of a function that takes four $n \times n$ matrices as input (A,B,C,D) and returns A*B + C*D? To answer this, consider only iterative approaches to matrix operations that a child in grade school would perform by hand.

- State the time complexity of F, where F is a function that takes in a list of sublists, and will return a list of sublists, where each sublist in the result is the sorted version of the corresponding sublist of the input assuming:
  - **(3 points)** You will use selection sort.

  - **(3 points)** You will use merge sort.

9. Consider the following relationships: $A \to B$, $B \to C$, $C \to E$, $E \to B$, $A \to L$, $L \to M$, $M \to N$, $N \to L$, $M \to A$; $x \to y$ indicates that there is a directed edge from x to y. For the following questions, ensure that you indicate your answers clearly, and show your work.

   - **4 points** Draw this graph.

   - **6 points** How many distinct strongly connected components (SCC) are in this graph? For each SCC, list the vertices that compose that SCC. Make sure your answer clearly separates the sets according to its SCC.

   - **4 points** Fully characterize the connectivity between A and E showing your evidence.

   - **4 points** Fully characterize the connectivity between A and N showing your evidence.

   - **5 points** Carefully list all cycles in the graph.

10. Consider the following relationships that characterize an undirected graph: A/B, A/C, A/E, E/C, B/D, C/D, D/F, F/G, D/G, E/G; ζ/ξ means that there is an edge between ζ and ξ. For the below, where relevant, when considering multiple vertices at the same level that you need to visit, choose the vertex with the lowest alphabetical value first. You must use the algorithm we presented in class and the lab. Clearly show all work.

   - **4 points** Draw this graph

   - **5 points** List the vertices that would be visited by a breadth-first traversal of this graph starting at D.

   - **5 points** List the vertices that would be visited by a depth-first traversal of this graph starting at C.

11. **10 points** An AVL tree is fed data in the following order (in order of ascending time): A,C,Y,P,M,N. $x < y$ means that x occurs before y in the English alphabet. Starting from an empty AVL tree, show how the tree diagram looks for **each** step; i.e., before each item is added, after the item has been added but before the AVL tree is restored, and after the item has been added and the AVL property restored. For full points, the AVL imbalance **must** be shown for all non-leaf nodes.

12. A min heap is fed data in the following order (in order of ascending time): 20,5,2,10,-2.

   - **10 points** Without skipping or combining any steps, show the construction of this heap as elements are added to it.

   - **2 points** For the completed heap, clearly show how the array that stores the heap will look, labeling the indices of the array using the C/Python convention.

   - **4 points** Without skipping or combining any steps, start with the final populated heap above, and show how four elements would be extracted. Show all steps, including each step of any heap adjustments that occur.

13. **6 points** Write a hash function in C (using the subset we covered in class) that will map the set of A (a subset of the integers) into the range $[5, 10]$. A is defined so that each element, i, of A, obeys $40 < i < 100$. Your hash function must properly account for the various ranges above (i.e., it can not be over-simplified, or degenerate).

14. **6 points** Write a simple, non-recursive C function that has $\mathcal{O}(log(n))$ complexity; you may assume n is an argument.

15. **6 points** Write a simple, non-recursive C function that has $\mathcal{O}(n^2 log(n))$ complexity; you may assume n is an argument.

16. Consider the graph represented by the adjacency matrix (where 0 represents the absence of an edge):

	a	b	c	d
a	0	11	5	0
b	11	0	8	8
c	5	8	0	9
d	0	8	9	0

- **4 points** Draw this graph and characterize it.

- **10 points** Draw all spanning trees where vertex A is connected to only one other vertex; quantify the weight of each spanning tree.

- **2 points** Identify the minimum spanning tree.

17. **10 points** Show clearly (as described below) how merge sort will sort the list X, A, C, Y, G, F, D, K, E in ascending order. Draw a tree diagram, where the nodes are lists that correspond to the sublists that are generated by the algorithm; draw this tree all the way down to the singleton leaves. Then draw an inverted tree (leaves at the top, root at the bottom), that shows how the singleton nodes are re-assembled. The root (at the bottom) will be the resultant sorted list.

18. **20 points** Consider the following graph relations: $(A/B, 6)$, $(A/C, 1)$, $(C/B, 2)$, $(B/D, 1)$, $(C/D, 5)$; $(\sigma/\epsilon, \xi)$ means that the edge connecting σ and ϵ has weight ξ. Showing all steps clearly, skipping nothing, perform Dijkstra's shortest path algorithm on this graph starting from A. Where relevant, use u and ∞. Clearly indicate your final answer (the output of Dijkstra's algorithm).

(You may use this page for rough work, but answers must appear on the page where the question was asked.)