ICLICKER QUESTIONS FOR DATA STRUCTURES TEST 3

-------------------------------------------------------

All solutions are at the end.

(I'd recommend copying them into a separate file to check as you go, or else write down all your answers and check them at the end.  If anyone can figure out a better way to format them, that would be awesome.)

As usual, these are accurate to the best of my knowledge, but I provide no guarantees and accept no responsibility for anything at all related.

-------------------------------------------------------

iClicker question 15.1: (Collaborative, Revisited)

What is the order notation for the number of operations in our non-linear word search algorithm, assuming s = search word length, and h \* w = dimensions of the board?

(A) O(8•s•w•h)

(B) O((s•w•h)^8)

(C) O((8•w•h)^s)

(D) O(w•h•8^s)

(E) O(w•h•s^8)

iClicker question 15.2: (Collaborative)

What is the average case order notation for the running time of Quicksort?  What is the average case order notation for the additional memory use (excluding the input vector and stack memory) or Quicksort?

(A) Running time: O(n^2) Memory: O(n)

(B) Running time: O(n log n) Memory: O(log n)

(C) Running time: O(n) Memory: O(n log n)

(D) Running time: O(n log n) Memory: O(1)

(E) Running time: O(n) Memory: O(1)

iClicker question 15.3: (Individual)

What's the first thing you would tackle in implementing a Game of Life simulation?

(A) How to represent the data/the board?

(B) I'm worried about the performance.  How will the simulation scale to

   large examples?  Will it run fast enough?

(C) Figuring out the user interface.  I'd like to make something web-based or

   an iOS or Android app… And it's got to be pretty.  Have to pick pretty

   colors!

(D) How will the program take input?  From a file?  Or can we use the mouse?

   Can the user switch the simulation speed?

(E) Find the power switch of the computer.  We'll need to reboot a lot to get

   this program debugged.

iClicker question 16.2: (Collaborative)

Which of the following statements is least true?

(A) In C++, integer variables are never initialized to 0.

(B) STL Maps are magical!

(C) STL Maps are like Python dictionaries.

(D) If you can code something with fewer keystrokes it is always better

   software.

(E) If your data contains duplicates and/or you don't want it sorted (you

   need it in a specific non-sorted order), you probably want an STL vector

   (or list), STL maps probably won't be helpful.

iClicker question 16.3: (Collaborative)

Which of the following is false for the STL pair class?

(A) Pairs are cool, they can be used to return to values from a function

   (more intuitive than that weird trick using pass-by-reference arguments).

(B) I don't need to put #include <utility> if I'm using an STL map in the

   same file (because it is indirectly included with #include <map>).

(C) The first item in an STL pair is always const and cannot be changed.

(D) first and second are public member variables of the STL pair struct,

   therefore they can be accessed and edited directly by the user (we don't

   need to use a get or set member function).

(E) For a previous homework I have (or could have easily) written my own STL

   pair-like class glueing together two related items.

iClicker question 16.4: (Collaborative)

Which of the following is true for the STL map iterators?

(A) Data is accessed in the order it was inserted.

(B) Visiting every element in an STL map is faster than visiting every

   element in an STL vector.

(C) Since STL map has an operator[] it is like STL vector.

   I can move a map iterator forward not just one spot (using itr++),

   but I can also jump forward an arbitrary number of spots (e.g., 25) using

   itr + 25.

(D) STL vector, STL list, and STL map iterators are all implemented as

   simple, direct pointers to an element in the container.  If an iterator

   is dereferenced it simply returns the value of the element.

(E) None of the above.

iClicker question 16.5: (Individual)

What do you think of STL map?  (Choose the statement that best describes your thoughts now.)

(A) Maps are magical.

(B) I'm sure maps will be helpful in HW6.  Can I use an STL map on HW6?

(C) Can we pretty please do the maps homework (HW7) instead of the

   super-scary, worse-than-impossible, did-I-miss-the-drop-deadline,

   just-cuz-its-painting-doesn't-make-it-fun homework (HW6)?

(D) I wonder how these map things are implemented… will we talk about that

   soon?

(E) There was a surprising amount of info in this short 4 page lecture

   handout… I need time in lab to practice this and really learn it.

iClicker question 17.2: (Collaborative)

Which of the following statements is false?

(A) Any C++ type can be used as the value (second) part of an STL map.

(B) Any C++ type can be used as the key (first) part of an STL map.

(C) Duplicate values are allowed in an STL map.

(D) Duplicate keys are not allowed in an STL map.

(E) operator< must be defined for the key type of the STL map.

iClicker question 17.3: (Collaborative)

What is the order notation to look up student X's grade in course Y?  Assume s students, c different classes offered at the school, each student takes up to k classes, and at most p students take a particular course.

(A) O(log p + k)

(B) O(log s \* k)

(C) O(p + log c)

(D) O(log s + k)

(E) O(s \* log k)

iClicker question 17.4: (Collaborative)

What is the order notation to make a list of all students who have taken course Y?  Assume s students, c different classes offered at the school, each student takes up to k classes, and at most p students take a particular course.

(A) O(s + log k)

(B) O(log s + k + p)

(C) O(s \* k)

(D) O(p \* k)

(E) O(s \* k + p)

iClicker question 17.5: (Collaborative)

Which of the following statements is false about a binary search tree holding the integers 1-10?

(A) The tree will have at least four "levels".

(B) If '6' is the left child of the root, then '2' cannot be the root node.

(C) If '5' is not in the left subtree of '7', then '7' must be in the right

   subtree of '5'.

(D) If '1' is the root node, then '10' cannot be a child of the root node.

(E) If the root is even, and the sum of the nodes in the left subtree is

   even, then the sum of the nodes on the right subtree must be odd.

iClicker question 18.1: (Collaborative)

How many exactly balanced binary search trees exist with the numbers 4.5, 9.8, 3.5, 13.6, 19.2, 7.4, 11.7 ?  How many exactly balanced binary trees exist with these numbers?

(A) 7! 2^7

(B) 1 7!

(C) 2^7 7!

(D) 1 2^7

(E) 42 42

iClicker question 18.2: (Collaborative)

Which of the following statements about STL container types is most true?

(A) A program that uses an STL set can easily be changed to use an STL map

   instead, with negligible performance impact.

(B) A program that uses an STL set can easily be changed to use an STL list

   instead, with negligible performance impact.

(C) A program that uses an STL map can easily be changed to use an STL set

   instead, with negligible performance impact.

(D) A program that uses an STL map can easily be changed to use an STL vector

   of STL pairs instead, with negligible performance impact.

(E) A program that uses an STL vector can easily be changed to use an STL

   list instead, with negligible performance impact.

iClicker question 18.3: (Collaborative)

What is the order notation of the find\_smallest function we just wrote, assuming the tree has n nodes in it and a height h?

(A) O(n)

(B) O(log n)

(C) O(log h)

(D) O(h)

(E) None of the above.

iClicker question 18.4: (Collaborative)

What is the order notation of the count\_odd function we just wrote, assuming the tree has n nodes in it and a height h?

(A) O(n)

(B) O(log n)

(C) O(log h)

(D) O(h)

(E) None of the above.

iClicker question 18.5: (Collaborative)

What is the height (# of levels) of the binary search tree that has post order traversal: 1 2 3 4 5 6 7

(A) 3

(B) 4

(C) 5

(D) 6

(E) 7

iClicker question 19.1: (Collaborative)

What is the traversal order of the destroy\_tree function we wrote earlier?

(A) in-order

(B) pre-order

(C) out-of-order

(D) post-order

(E) breadth-first

iClicker question 19.2: (Collaborative)

What is the sum of the last 4 elements in a breadth first traversal of an exactly balanced binary search tree with the elements 1-7?

(A) 10

(B) 16

(C) 22

(D) 28

(E) None of the above.

iClicker question 19.3: (Collaborative)

For the breadth-first traversal code we just wrote…

What is the best/average/worst-case running time?

What is the best/average/worst case additional memory usage?

Give a specific example tree that illustrates each case.

(A) Running time: Best: O(1) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(n) Average: O(n) Worst: O(n)

(B) Running time: Best: O(n) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(log n) Worst: O(n log n)

(C) Running time: Best: O(1) Average: O(log n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(log n) Worst: O(n)

(D) Running time: Best: O(n) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(n) Worst: O(n)

(E) Running time: Best: O(1) Average: O(log n) Worst: O(log n)

   Memory Usage: Best: O(1) Average: O(1) Worst: O(1)

iClicker question 19.4: (Collaborative)

Consider a BPlusTree node with b = 5 and keys: "ant", "bear", "bee", "cod";

If a search is looking for the key "cat" and the node is not a leaf, counting from the left, which child should the search use?

(A) The first child (children[0])

(B) The second child (children[1])

(C) The third child (children[2])

(D) The fourth child (children[3])

(E) The fifth child (children[4])

iClicker question 19.5: (Collaborative)

Consider a BPlusTree node with b = 4.  What is the maximum number of nodes that be on level i?

(There's subscripting or superscripting in plaintext, so ^ is superscript and v is subscript.)

(A) log v(b) i

(B) log v(i) b

(C) b^(i)

(D) i^(b)

(E) b \* i

iClicker question 20.2: (Collaborative)

Which of these statements is true about B+ trees?

(A) B+ trees were created just to torture computer science students.  They're

   complicated and we didn't even discuss erase()!

(B) B+ trees are a good choice for when a BFS search should be fast.

(C) B+ trees are a good choice when getting a node takes a long time, but

   things are fast once the node is in memory.

(D) Since B+ trees tend to be much wider than a binary search tree, they

   would be a bad choice for our ds\_set class.

(E) They're called B+ trees because they're such a bad idea that when one

   student proposed the data structure, their professor gave them a B+ in

   the course.

iClicker question 20.3: (Collaborative)

For the tree with n nodes, for the recursive tree height algorithm we just wrote…

What is the best/average/worst-case running time?

What is the best/average/worst-case memory usage?

Give a specific example tree that illustrates each case.

(A) Running time: Best: O(n) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(log n) Average: O(log n) Worst: O(n)

(B) Running time: Best: O(n) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(log n) Worst: O(n log n)

(C) Running time: Best: O(n) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(1) Worst: O(1)

(D) Running time: Best: O(1) Average: O(log n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(n) Worst: O(n)

(E) Running time: Best: O(n) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(log n) Average: O(log n) Worst: O(log n)

iClicker question 20.4: (Collaborative)

For the tree with n nodes, for a breadth-first shortest path to leaf node algorithm…

What is the best/average/worst-case running time?

What is the best/average/worst-case memory usage?

Give a specific example tree that illustrates each case.

(A) Running time: Best: O(log n) Average: O(log n) Worst: O(log n)

   Memory Usage: Best: O(log n) Average: O(log n) Worst: O(log n)

(B) Running time: Best: O(1) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(n) Worst: O(n)

(C) Running time: Best: O(1) Average: O(n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(1) Worst: O(1)

(D) Running time: Best: O(1) Average: O(log n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(log n) Worst: O(n)

(E) Running time: Best: O(1) Average: O(log n) Worst: O(n)

   Memory Usage: Best: O(1) Average: O(1) Worst: O(1)

iClicker question 20.5: (Collaborative)

Which of the following statements about tree iterators is false?

(A) Sometimes the "next" node in the tree is O(log n) steps away!

(B) If the tree iterator is pointing at the node containing the last element

   in sorted order, that node must be a leaf node.

(C) When we are part way through an iteration of a tree using the stack of

   pointers method, and someone calls insert or erase on that tree we must

   assume the iterator is now invalid.

(D) Once we have a fully debugged operator++ for tree iterators, we can

   simply swap all the lefts & rights to implement operator--.

(E) None of the above.

iClicker question 20.6: (Collaborative)

For either version of the tree iterator operator++ function, for a balanced tree with n elements what is the order notation for:

the worst case single call to operator++,

the average (or amortized) single call to operator++,

the total running time for a loop increment from begin() to end().

(A) Single worst: O(1) Single average/amortized: O(1)

Total: O(n)

(B) Single worst: O(log n) Single average/amortized: O(log n)

Total: O(n)

(C) Single worst: O(log n) Single average/amortized: O(log n)

Total: O(n log n)

(D) Single worst: O(log n) Single average/amortized: O(1)

Total: O(n)

(E) Single worst: O(n) Single average/amortized: O(log n)

Total: O(n log n)

iClicker question 21.1: (Collaborative)

Which of the following statements about operator overloading is true?

(A) The mathematical operations +, -, \*, and / can only be applied to data of

   integer or floating point type.

(B) The function operator- can only be implemented as a binary (2 argument)

   operator.

(C) When a operator is written as a non-member function, it can still have

   access to the private member variables and functions of the class.

(D) If a member function takes in as an argument a second object of the class

   type, it has access to the private member variables of both the "this"

   object and the argument object.

(E) All of the above.

iClicker question 21.2: (Collaborative)

( Class did not get this far. )

(A)

(B)

(C)

(D)

(E)

ANSWERS:

Please make sure to compare the question numbers, since most polls are not in the question section but are in the answer section.

15.1: D 15.2: D 15.3: poll

16.1: poll 16.2: D 16.3: C 16.4: E 16.5: poll

(16.4 should not appear on test.)

17.1: poll 17.2: B 17.3: D 17.4: C 17.5: C/D

18.1: B 18.2: A 18.3: D 18.4: A 18.5: E

19.1: D 19.2: B 19.3: D 19.4: D 19.5: C

(19.5 may be modified on exam.)

20.1: poll 20.2: C 20.3: A 20.4: B 20.5: B,C  20.6: D

21.1: D