|  |
| --- |
| EECE 2080C-001 - Fall 2018 - Final Exam  University of Cincinnati |

Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ MNumber:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instructions

* Please read through this entire exam very carefully before starting.
* This exam is closed notes and closed electronic devices. You may use a single sheet of paper with anything you want on it and your text book. Your single sheet of paper must be turned in with the exam.
* All work must be written on the exam pages in order to be graded. Any scrap paper used, must be the scrap paper provided during the exam period.
* For programming questions: Please be accurate with your C++ syntax: this includes appropriate use of braces, semicolons, and the proper use of upper/lowercase letters. I will deduct points specifically because of syntax errors such as missing semicolons but any confusion from your code will be resolved by a compiler. If it doesn’t compile, I reserve the right to not change partial credit assigned.
* No electronic devices may be used during the exam unless permitted in advance: this includes (but is not limited to) calculators, phones, tablets, drones, Amazon Echo, and computers.
* You have 120 minutes to complete the exam.
* Unless otherwise stated, each question (1-41) is worth 2 points. The coding problem at the end is worth 30 points. The test has a maximum of 124 points available but scoring 120 points will be considered 100%. Points above 120 are extra credit.
* If you desire to have your graded exam back, I will make them available in the department office for the first two weeks of the summer semester.

Good luck!

Hash Table

Implementation options?

* + dictionary, giant linked list or giant array, BST, best would be using hashing

What is a clash?

* When two objects wind up with the same hash key

How to deal with Clashes (probably a diagram)

* Linear – what we did for lab
* Quadratic (i^1, i^2, i^3 etc.)
* linked list – itd be a lot like an adjacency list; see drawing

Utilization factor

* how full the table is/the density of the table
* why do we care?
  + It takes longer and longer once over 2/3 utilized

What impacts look up time?

* Do we want a complicated hash that makes unique values or do we want a fast one that sometimes gives you clashes
  + Better to be fast in most cases, for look up

Perfect Hash needs what?

* Need a 1:1 mapping from hash to hashtable; need to know every value youre going to put in

What makes a good hash function?

* Fast
* Spreads values out well/good distribution

Development vs production; dev is working well and production is clashing: likely cause? You didn’t test real/similar enough data in dev to whats going in in production

Heap

What ADT is Heap close to?

* BST

Similar and what is different?

* Sim:
  + Both tree shaped
* Dif:
  + heap only has min or max at the top
  + heap is always complete (starts at root, and progresses to children repeatedly)
  + tree can look like a big linked list if not balanced

Know how to draw an insert or delete – drawn in class

What ADT is best for:

Fast retrieval Only of values?

* Hashtable because lookup should be quick unless theres lots of clashes

Modifying values often?

* Hashtable still

Space vs. Time?

* Don’t care about space: Hash
* Care about space: array using binary search

Summary report is needed of data (traversal of values)

* Sorted array/vector

Pointer based implementation vs array pros and cons

* Pointer – dynamic size
* Array – direct access

Binary Search Tree

Child \ parent relationship (right child and left child)

* Typically left child is less than parent and right is greater

For a full tree how many nodes in tree, how many at a level



At a particular level how many nodes can there be?

* Nodes = 2^level # (starting a zero)
  + Example: level 2 would have 4 nodes on that level and 7 nodes from there and above

What is a balanced tree?

* The left and the right sub trees are within a certain height tolerance of one another

And why is it desirable?

* Unbalanced turns into a linked list

Root – just one in a tree

Unix commands

What is the dir equivalent ?

* Ls (lists the files there)
* (in his example, he put ls -l ; I think that was the target directory?)

What is the option to get all long listing?

* Ls - l

What is a Makefile?

* It runs the command compile your program

Chmod <mask> 755 , 644, 777 - <https://en.wikipedia.org/wiki/Chmod>

* Changes the privileges for user, group, world

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Permission** | **rwx** | **Binary** |
| 7 | read, write and execute | rwx | 111 |
| 6 | read and write | rw- | 110 |
| 5 | read and execute | r-x | 101 |
| 4 | read only | r-- | 100 |
| 3 | write and execute | -wx | 011 |
| 2 | write only | -w- | 010 |
| 1 | execute only | --x | 001 |
| 0 | none | --- | 000 |

Sorting Algorithms

* Will be about 5 elements
* Radix sort
* Merge Sort
* Quicksort - <http://me.dt.in.th/page/Quicksort/>

Priority Queue vs. Heap

1. Define a class that you can use as a node in a double linked list that stores a previously defined structure named Product as its data. (each node has a next and a previous pointer; has a head and a tail)

Priority keeps a queue/list with items with the highest priority; most basic returns just top

Why PQ over heap – want more than just the top item (say top 5, example)  
  
Stack and Queue – kind of like the questions on last exam but not drawn this time

Underflow error (empty) – when you try to pop but its empty

Overflow error (full) – it’s full  
  
Doubly Linked List

-Like Linked List but Node is different, what is different? – has prev and next pointers

Graphs

* Directed, undirected,
* Weighted unweighted
* Adj matrix implementation
* Adj list implementation
* DFS and BFS? Kevin Bacon number
* MST
* Prim’s algorithm

Tree

* Inorder, pre-order, and post order traversal
* Has root and kids
* Typical is a BST

What is the difference between the dot and arrow operators?

* Dot - is called on objectes
* Arrow – is called on pointers

Destructors/constructors

* Should know

Virtual methods

* Whats it mean and why do we do this – do polymorphism for objects where behavior is determined at run time rather than compile time (think the vehicle example); without it we would need to dynamic cast which is more expensive. This reuses code better.
* pure abstract – put virtual in the base class next to the base class method and override for the individual, sub classes

Big O times for search (linear, BST)

A code question,

You will need to define a class, and it’s members (not give the implementation for those members) though. And then asked to solve a problem.

Write the code to search a directed graph stored in an Adjacency List to determine if two nodes are connected by exactly one other node. For example, A->B->C then OneStep(A, C) would return true. This should include a class declaration and the definition for all required members that are needed to support the search. You don’t need to code those members. You can assume any other data structures you wish to use such as a stack, heap, or linked list are defined elsewhere and you can use standard functions without declaring or defining them.