# University of Nevada, Las Vegas Computer Science 477/677 Spring 2022

## Answers to Examination April 20, 2022

The entire examination is 330 points.

1. True or False. [5 points each]

   (a) **F** Open hashing uses open addressing.

   (b) **F** Kruskal's algorithm used dynamic programming.

   (c) **F** There will be no collisions if the size of a hash table is at least ten times the number of data items.

   (d) **T** A hash function should appear to be random, but cannot actually be random.

2. Fill in the blanks. [10 points each]

   In problems (a) and (b), let $n$ be the number of vertices, $m$ the number of arcs, and $p$ the maximum number of arcs in the shortest path between any two vertices.

   (a) The asymptotic complexity of the Floyd/Warshall algorithm is $\Theta(n^3)$

   (b) The asymptotic complexity of Dijkstra's algorithm algorithm is $O(m \log n$.

   (c) A **perfect** hash function fills the hash table exactly with no collisions.

   (d) **Huffman's** algorithm finds a binary code so that the code for one symbol is never a prefix of the code for another symbol.

   (e) **Huffman's** and **Kruskal's** are greedy algorithms that we've studied this semester.

   (f) An acyclic directed graph with 9 vertices must have at least **9** strong components. (Must be exact answer.)

   (g) In **separate chaining** or **open hashing** there can be any number of items at a given index of the hash table.

   (h) The asymptotic expected time to find the median item in an unordered array of size $n$, using a randomized selection algorithm, is $\Theta(n)$.

   (i) If $h(x)$ is already occupied for some data item $x$, a **probe sequence** is used to find an unoccupied position in the hash table.

3. Give the asymptotic complexity, in terms of $n$, for each of these code fragments. [10 points each]

   (a)
   ```
   for(int i = 0; i < n; i++)
     for(int j = n; j > i; j = j/2)
   ```

   $\Theta(n)$

   (b)
   ```
   for(int i = 0; i < n; i++)
     for(int j = i; j > 0; j = j/2)
   ```

   $\Theta(n \log n)$

4. Solve each recurrence, giving asymptotic answers, using $O$, $\Omega$, or $\Theta$, whichever is most appropriate. [10 points each]

   (a) $F(n) \leq 4F(n/2) + n^2$

   $F(n) = O(n^2 \log n)$

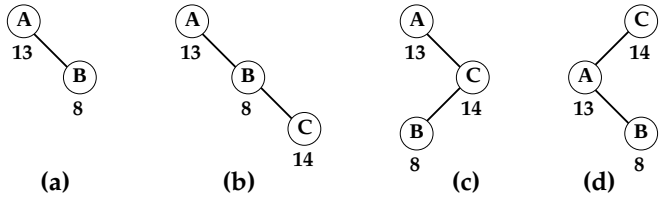
   (b) $G(n) \geq G(4n/5) + G(3n/5) + n^2$

   $G(n) = \Omega(n^2 \log n)$

   (c) $T(n) = T(3n/10) + T(n/5) + n$

   $T(n) = O(n)$

5. [20 points] You need to store the items A, B, and C, in that order, in a treap. The random key for A is 13, for B is 8, and for C is 14. Use maxheap order. Draw the resulting treap after each insertion, and show each rotation.




   We first insert A, then B. Heap order is preserved, so no rotation is necessary.

   We next insert C. To restore heap order, we do a left roatation at B.

   Heap order is still not restored. We do a left rotation at A. Heap order is then restored.

6. [20 points] The BFPRT selection algorithm has asymptotic time complexity $\Theta(n)$, which is proved using a recurrence. Give that recurrence.

   $T(n) = T(3n/10) + T(n/5) + n$

7. [30 points] Consider the function $F$ computed by the recursive code given below.

   (a) What is the asymptotic complexity of $F(n)$?
   $F(n) = \Theta(n^2)$

   (b) What is the asymptotic time complexity of the recursive code when it computes $F(n)$?
   $T(n) = 2T(n/3) + 1$
   $T(n) = \Theta\left(n^{\log_3 2}\right)$

   (c) What is the asymptotic time complexity of a memoization algorithm which computes $F(n)$?
   $T(n) = \Theta(\log n)$

```
int F(int n)
 {
  if(n < 3) return 1;
  else return F(n/3)+2*F((n+1)/3)+n*n;
 }
```

8. [20 points] If the array $A[5][7]$ is stored in column-major order, how many predecessors does $A[3][4]$ have?

   The predecessors are $A[0][0] \ldots A[4][3]A[0][4] \ldots A[2][4]$. There are 20 in the 2D block and 3 in the 1D block, for a total of 23.

9. [20 points] Explain how to implement a sparse array using a search structure. Hint: You must explicitly describe how to implement the operators **fetch** and **store**.

   Call the sparse array $A$. We write 0 for the default value. Use a search structure $S$. Each item in $S$ is an ordered pair $(i, x)$, where $A[i] = x$, and $i$ is the key. To implement fetch, we execute findA$(i)$. If the pair $(i, x)$ is found, return $x$. If the find fails, return 0. To implement store$(i, x)$, we first execute findA$(i)$. if no pair is found, insert the pair $(i, x)$ into $S$. Otherwise, if the find return the pair $(i, y)$, replace $y$ by $x$ in that pair.

10. [20 points] You are implementing a 3D triangular array $A$ where $A[i][j][k]$ is defined for $i \geq j \geq k \geq 0$, as a one-dimensional subarray of main memory, and you wish to store $A$ in row-major order, with base address 1024. where $A[i][j][k]$ is defined for $i \geq j \geq k \geq 0$, Each term of $A$ takes one place in main memory. What would be the address, in main memory, of $A[7][4][3]$?

    Since this problem is on the homework, I will not give the solution here.

11. [20 points] You are given an acyclic directed graph $G = (V, E)$ where each arc is weighted. If $(x, y)$ is an arc, we write $w(x, y)$ for the weight of that arc. Describe a dynamic programming algorithm which calculates the directed path through $G$ of maximum weight. (The weight of a path is defined to be the sum of the weights of its constituent arcs.) (Hint: your algorithm should use words such as "in-neighbor" or "out-neighbor.")

    Since this problem is on the homework, I will not give the solution here.

12. [20 points] Walk through the first two steps of Johnson's algorithm for the weighted directed graph shown in (a) below. In (b), show the values of the function $h$, where $h(v)$ is the shortest distance from the fictitious source to $v$. In (c), show the adjusted weights of the edges.