CS310: Advanced data structures and algorithms

Midterm Exam – practice

Instructor – Nurit Haspel

General Instructions

1. You may use any printed/written material. Electronic devices are not allowed.

2. The work is to be your own and you are expected to adhere to the UMass Boston honor system.

3. The exam contains five questions. The weight of each question is listed. Read each question carefully before you answer.

4. Write your answers in the available spaces, using the back of the page if needed. Write clearly and concisely and try to avoid cursive.

5. Please explain your answers if needed but do it briefly.

6. If you base your answer on a homework question or class notes state it in your answer.

Good Luck!

Name (as appears on your student ID): ________________________

<table>
<thead>
<tr>
<th>Question</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>
1. Runtime analysis (20%)

a. (12%) Given the following piece of code:

```java
public static int someFunction(int n)
{
    int i, j, sum = 0;
    for (i=0; i < n; i+=2) /* loop 1 */
        sum += i*i;
    for (i=0; i < n; i++) /* loop 2 */
        for (j=1; j < n; j=j*2) /* loop 3 */
            sum += i*j;
    return sum;
}
```

What is the run time of:

(a) Loop 3

(b) Loop 2 (including the runtime of loop 3).

(c) Loop 1.

(d) The whole function, explain briefly.

b. (8%) An algorithm takes 15 seconds to solve a problem of size 1000. If the algorithm is quadratic – i.e., runs as $O(N^2)$, how large a problem can be solved in 60 seconds?

(a) 2000
(b) 4000
(c) 6000
(d) none of the above
2. Java Data Structures (20%) 

For (a-c) below, determine what is the best data structure to use out of the ones we discussed in class: List, Set, Map. If more than one acceptable answer exists, use the most efficient one (with respect to runtime) that has the power you need. Also say what type you use: LinkedList vs. ArrayList, HashSet vs. TreeSet or HashMap vs. TreeMap. In the line inside the triangular brackets write the data type. For example, if you think that the correct answer to a question is a List of Integers, then write: 

List<__________> (and in the space below say whether it’s a linked or array list).

Please provide a brief explanation.

(a) (7%) You are working on a banking program. Each day, the checks for one account are processed and (assuming they don’t “bounce”) an appropriate Check object is added to the Account object, to wait for end-of-month processing to write the bank statement. The check objects have fields number (of type int), received (of type Date) and amount (of type Money) and must be kept in original order by time of arrival, and are only used once (in our simplified system) to write the statement, where they are reported in the same order.

List<__________>, Set<__________>, Map<__________, __________>

(b) (7%) In the same banking scenario, each account has a unique id (an Integer) and one or more owners identified by social security numbers (also ints). Also bank account holders may have several accounts, each with a unique id. Explain how you can use two Collections API classes working together to support looking up the bank accounts ids for given social security numbers.

List<__________>, Set<__________>, Map<__________, __________>

(c) (6%) You want to be able to display the courses a student took (represented as Strings, like “CS310”) and their grades (represented as Characters between ‘A’ and ‘F’, assume there are no “A-” grades etc.), sorted by the grade. Explain briefly.

List<__________>, Set<__________>, Map<__________, __________>
3. Hash tables (20%):

(a) (8%) You created a hash table using separate chaining and forgot to rehash. You then realized that you inserted $n \log(n)$ elements into the hash table whose array size is $n$. What would be the average lookup time for your hash table now, assuming a good hashing function? Explain briefly.

(b) (12%) We implement a hash table of integers, using an array of size 7. The hashing function for an integer $x$ is $h(x) = x \% 7$. We use linear probing to resolve collisions. The elements are inserted in the following order: 1, 15, 14, 3, 10, 5, 25. We do not rehash. Draw the final configuration of the table and determine how many collisions each element will cause in total. That is – if we have to explore two consecutive probe positions, it is two collisions etc. For your convenience you may use the following table and illustration of the hash table.

<table>
<thead>
<tr>
<th>Number</th>
<th>Hash value</th>
<th># collisions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. Induction (15%). Prove the following formulas. Don’t forget to state explicitly the
three stages:

- Base case
- Inductive hypothesis for $1 \leq k < n$
- The inductive step from $n - 1 \rightarrow n$ (or $n \rightarrow n + 1$ if it works better)

(a) For all $n \geq 1$, show that $1 + 4 + 7 + \ldots + (3n - 2) = \frac{n(3n - 1)}{2}$

(b) For all $n \geq 1$, show that $n^3 + 2n$ is divisible by 3. For your convenience,

$(n + 1)^3 = n^3 + 3n^2 + 3n + 1$. 
5. Graphs (25%)

(a) (7%) Remember that the degree of a vertex in an undirected graph is defined as the number of edges touching this graph. Given an undirected simple graph \( G = (V, E) \), what is the sum of all the degrees of all the vertices of \( G \)? (a simple graph is a graph where every two vertices are connected by at most one edge).

a. \( 2|V| \)
b. \( 2|E| \)
c. \( |V| \)
d. \( |E| \)

(b) (8%) Based on your answer to (a) above, prove that in any simple undirected graph, the number of vertices with odd degrees is even.

**Hint:** This is probably the easiest question you’ll ever be asked. Ever.

(c) (10%) Trace the run of Breadth-First search (BFS) algorithm starting from b in the graph below. For tracing, use the same notation as in the class notes. Do not show the queue, just the order in which the edges are explored. Mark an * near the edges that participate in the final tree. Also mark the distances of all the vertices from b.

![Graph](image)