1. Runtime analysis (from Midterm F16)

a. 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. 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
2. Java Data Structures (from Midterm F16): 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. Please provide an explanation.

(a) 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) 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) 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 (from Midterm F16):

(a) 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? Explain briefly.

(b) What (really bad thing) could happen to the insert/lookup time for a scenario as in (a) above if you used linear probing?
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. 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. Prove the following formulas. Don’t forget to state explicitly the three stages:

- Base case
- Inductive hypothesis for \( 1 < k < n \)
- 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 \( 1 + 4 + 7 + \ldots + (3n - 2) = \frac{n \times (3n - 1)}{2} \)

(b) For any positive integer \( n \), \( 6^n - 1 \) is divisible by 5.

5. Graphs:

(a) Remember that the degree of a vertex in an undirected graph is defined as the number of edges touching this vertex. Prove that in any simple undirected graph, the number of vertices with odd degrees is even. (a simple graph is a graph where every two vertices are connected by at most one edge).

**Hint:** This question has a VERY simple answer, please don’t over-think it. The answer follows from the definition of the degree of a vertex and the definition of an edge. 1-2 sentences will suffice. Also, remember that an example does not constitute a proof. If you can’t possibly get the answer, you can prove by induction on the number of edges in a graph. A correct proof by induction will still get you full marks.

(b) 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.

```
   a---c---f
   |     |     |
  b    e    d
```