CSC 212 Practice Final Exam
Problems marked with (*) are challenging and problems marked with (**) are hard

Your Name: ______________________________

1. (10 points) Insert $1, 10, 3, 13, 8, 0$ into an initially empty separate chaining hash table with capacity 7 and hash function $h(k) = 3k \bmod 7$. Draw the table after the final insertion. Assume insertions occur at the **front** of chains.

   Solution: The final table is $[[0], [], [8, 3, 10], [3], [13], [], [], []]$.

2. (10 points) Insert $1, 0, 7, 8, 14, 5, 12$ into an initially empty linear probing hash table with capacity 7 and hash function $h(k) = 3k \bmod 7$. Draw the table after the final insertion.

   Solution: The final table is $[0, 7, 14, 1, 8, 15, 12]$.

3. (10 points) (*) Design suitable hash functions $h_1(k)$ and $h_2(k)$ for a double hashing table with capacity 17.

Solution: One possible answer is $h_1(k) = k \bmod 17$ and $h_2(k) = 1 + (k \bmod 16)$.

4. (10 points) Consider a linear probing hash table with capacity m and load factor 0.5. In the worst case, how many probes are performed by `contains`?

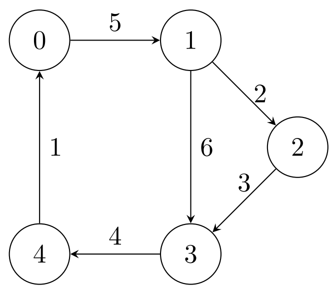
Solution: In the worst case, all $\frac{m}{2}$ full slots are inspected, and 1 empty slot, for a total of $\frac{m}{2} + 1$ slots.

5. (10 points) Convert the following adjacency matrix to an adjacency list.

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Solution: The adjacency list is $[[1, 2, 3, 4], [2], [2], [2], [0, 1, 2, 3]]$.

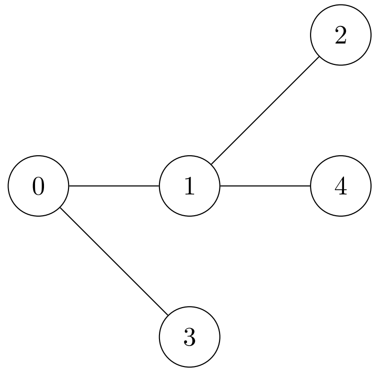
6. (10 points) Give the adjacency matrix for the following graph.



Solution: The adjacency matrix is

$$\begin{bmatrix} 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 2 & 6 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

7. (10 points) Find the lexicographically smallest depth-first traversal of the following graph.



Solution: The lexicographically smallest depth-first traversal is $0, 1, 2, 4, 3$.

8. (10 points) (**) Consider a social media platform where users can have **mutual friends**. That is, if user A is friends with user B, user B is also friends with user A.

   We say users A and B are **indirect friends** if A is friends with B or one of A's friends is an indirect friend of B. A **community** is a set of users where every pair of users are indirect friends and no users have indirect friends outside of the community.

   Describe how to model friendships as a graph, clearly defining the vertices and edges. Give pseudocode for computing the total number of communities.

Solution: The vertices are users. There is an edge between two vertices if and only if the corresponding users are friends. To compute the total number of communities

1. Create an empty set to track visited vertices, and an counter initialized to zero.
2. For each vertex v,
   (a) If v is already visited, skip it.
   (b) Otherwise, run a DFS starting at v that marks all traversed vertices as visited. Increment the counter by one.
3. Return the counter.

9. (10 points) (*) An **RL-rotation** is equivalent to a right rotation on the root's right child, followed by a left rotation on the root.

   Implement the `rl_rotate` function. Assume that an RL-rotation on the subtree rooted by `root` is always valid. Return the new root after rotation. Your implementation must run in $\mathcal{O}(1)$ time.

```
struct Node {
    Node* left;
    Node* right;
    // ...
};

Node* rl_rotate(Node* root) {
    // TODO: Implement this function.

}
```

Solution:

```
struct Node {
    Node* left;
    Node* right;
    // ...
};

Node* rl_rotate(Node* root) {
    auto z = root;
    auto x = z->right;
    auto y = x->left;
    auto b = y->left;
    auto c = y->right;
    y->left = z;
    y->right = x;
    z->right = b;
    x->left = c;
}
```

10. (10 points) Insert $0, 1, 2, 3, 4, 5, 6, 7$ into an initially empty red-black tree. Draw the resulting tree, including colors, after each insertion.

Solution: Check with `https://www.cs.usfca.edu/~galles/visualization/RedBlack.html`.