Quiz – 9 - Solution Set

Variant 1

Q1) Select one answer for each of the following Multiple-Choice Questions (MCQs): [2 x 4 = 8 pts]

1. D
2. D
3. C
4. B

Q2) **Bug:** only adds user->post interaction but not post->user interaction [1 point]

```
userToPostEdges[userID].push_back({postID, weight});          [1 point]

postToUserEdges[postID].push_back({userID, weight});           [1 point]
```

OR

```
userToPostEdges.at(userID).push_back({postID, weight});       [1 point]

postToUserEdges.at(postID).push_back({userID, weight});       [1 point]
```

Q3)

```
std::vector<User*> UserSearchEngine::searchByUsernamePrefix(const string& prefix){
        std::vector<User*> results;
        // Get all users sorted by username
        auto allUsers = usersByName.inOrderTraversal();              [1 pt]
        for (const auto& pair : allUsers) {
                if (pair.first.length() >= prefix.length() &&       [1 pt]
                pair.first.substr(0, prefix.length()) == prefix) {  [1 pt]
                        results.push_back(pair.second);             [1 pt]
                }
        }
        return results;
}
```

Q4) Bug: includes posts user already saw [2 pts]

Updated code:

```
vector<int> InteractionGraph::recommendPosts(int userID, int topN)
const
{
    vector<int> recommended;
    auto similar = findSimilarUsers(userID, 5);

    unordered_set<int> seen;
    for (auto& p : userToPostEdges.at(userID))
        seen.insert(p.id);

    for (auto& [other, score] : similar) {
        for (auto& p : userToPostEdges.at(other)) {
            if (!seen.count(p.id))                    // [2 pts]
                recommended.push_back(p.id);
        }
    }

    return recommended;
}
```

Q5) Accept any answer [1 point]

Variant 2

Q1) Select one answer for each of the following Multiple-Choice Questions (MCQs): [2 x 4 = 8 pts]

1. E
2. D
3. D
4. A

Q2) Bug: includes posts user already saw [2 pts]

Updated code:

```
vector<int> InteractionGraph::recommendPosts(int userID, int topN)
const
{
    vector<int> recommended;
    auto similar = findSimilarUsers(userID, 5);

   unordered_set<int> seen;
    for (auto& p : userToPostEdges.at(userID))
        seen.insert(p.id);

    for (auto& [other, score] : similar) {
        for (auto& p : userToPostEdges.at(other)) {
            if (!seen.count(p.id))                    // [2 pts]
                recommended.push_back(p.id);
        }
    }

    return recommended;
}
```

Q3) Bug: edge added in only one direction / missing other direction [1 pt]

```
void GeographicNetwork::addEdge(int from, int to, int weight)
{
    adjList[from].push_back({to, weight});
    adjList[to].push_back({from, weight});    // [2 pts]
}
```

Q4) bool CategoryNode::removePost(Post* post) {

```
    auto it = find(posts.begin(), posts.end(), post);

    if (it != posts.end()) {

        posts.erase(it);

        if (categoryName == "root"){
            updatePostCounts();                 [1 pt]
        }
        else{
            shared_ptr<CategoryNode> curr = parent;
            while(curr->categoryName != "root"){        [1 pt]
                curr = curr->parent;                    [1 pt]
            }
            curr->updatePostCounts();                   [1 pt]
        }
        return true;
    }
    return false;
}
```

Q5) Give points for any TA ‘s name