Section #3

Ashley Taylor and Shreya Shankar (Based on handouts by previous CS106B instructors and TAs)

Sets, Maps, and Recursion: Solutions

1. Remove Duplicates (Sets)

```
void removeDuplicates(Vector<int>& v) {
    Set<int> set;
    for (int i = 0; i < v.size(); i++) {
        if (set.contains(v[i])) {
            v.remove(i);
            i--;
        } else {
            set.add(v[i]);
        }
    }
}
```

2. Friend List (Maps)

```
Map<string, Set<string> > friendList(string filename) {
    // open file
    ifstream input;
    input.open(filename.c_str());

    // read file data into map of sets
    Map<string, Set<string> > friends;
    string name1, name2;
    while (input >> name1 >> name2) {
        friends[name1].add(name2);
        friends[name2].add(name1);
    }

    return friends;
}
```

3. Recursion Mystery

8

4. Print Stars (Recursion)

```
void printStars(int n) {
    if (n > 0) {
        cout << "*";          // print one star myself
        printStars(n - 1);    // recursion to do the rest
```

```
    }
}
```

4. Star String (Recursion)

```
string starString(int n) {
    if (n < 0) {
        throw n;
    } else if (n == 0) {
        return "*";
    } else {
        string s = starString(n - 1);
        return s + s;
    }
}
```

6. Sum of Squares (Recursion)

```
int sumOfSquares(int n) {
    if (n < 0) {
        throw n;
    } else if (n == 0) {
        return 0;
    } else {
        return n * n + sumOfSquares(n - 1);
    }
}
```

7. Subsequence (Recursion)

```
bool isSubsequence(string big, string small) {
    if (small == "") {
        return true;
    } else if (big == "") {
        return false;
    } else {
        if (big[0] == small[0]) {
            return isSubsequence(big.substr(1), small.substr(1));
        } else {
            return isSubsequence(big.substr(1), small);
        }
    }
}
```