Group Assignment 2

Lance Umagat and Garett Pement

Section 1: Introduction

There are N lockers and there are T tennis balls in the lockers. Each locker has a specific key that will unlock it with shared locks having the adjacent keys. So if key K unlocks locker N then contains the Keys for lockers N+1 and N-1. Although if Locker N is either the first or last locker then they will only have keys for N+1 or N-1 respectively. The problem we are trying to solve is how to collect all the tennis balls while using the least number of keys i.e. find the most optimal route.

Section 2: Algorithm 1

Sub section 1:

Lockers.sort()

All\_lockers = [set of X in range of number of lockers +1]

For keys in key\_set:

Key.sort()

Path = count\_path(key,locker,desired)

Total.append(path)

Return min(total opened)

Sub section 2:

First sort the lockers. Then create an array that will hold all of the lockers up to an endpoint. Check keys in a key set and sort the keys for I to number of keys. Then we recursively call a path function that counts the path for that key. Lastly we will use an array to append the total opened using the path variable.

Sub section 3:

Sub section 4:

Section 2: Algorithm 2

Sub section 1:

If I == 0;

Print keys[i];

Return left\_path(keys[i], all\_lockers) +1

If I == len(keys) – 1;

Return right\_path(keys[i], all\_lockers)

Else

Return total\_midpath(keys[i-1], keys[i], all\_lockers) + functioin(i-1, keys, all\_lockers) +1

Sub section 2:

If the key is the initial key, then print the left path and add one to it. If the length of the keys is at the other end of the array (last locker), then print all of the keys used (right path). Else, return the total mid path that will check up, left, right, and diagonal, and call the function recursively .

Sub section 3:

Sub section 4: