TripletSum Project

Robert Frenken

CSE 2331

This code implements a hash map (dictionary in Python) that will create a unique key and value for every set of 2 values in the integer list. It will then iterate through the list again, and with the hash map determine if the resulting sum equals sumVal. If so, it will be added to a list, and the first element of the list will be the output. If there is no such triplet, the output will state that it couldn't find a triplet. This runs in approximately O(n^2) time, as there are 2 separate nested for loops. Using a hashtable reduces the time complexity, as if everything was nested into loops, the time complexity would be O(n^3). The input section runs in O(n) time, as it has to read the data file.

Pseudocode:

```
dataX <- input from user

sumVal <- input from user

hashtable <- new hashtable

for element1 in dataX do:
        for element2 in dataX do:
                if (element1 != element2):
                        result <- element1 + element2
                        hashtable.add({(element1, element2,), result})
listOfSolutions <- new list

for k in dataX do:
        for key in hashtable do:
                I, j <- key[0], key[1]
                If !(i == k or j == k):
                        result <- k + hashtable[key]
                        if result == sumVal:
                                listOfSolutions.add(I, j, k)
                                break
if listOfSolutions.size > 0:
        print(listOfSolutions[0])
else:
```

```
print(“No solution”)
```