Sorting it all out

LAB # 6

By

Corey Henry

And

Natalie Morrison

***“On my honor, as a Mississippi State University student, I have neither***

***given nor received unauthorized assistance on this academic work.”***

Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CSE-1384-06-201430 Intermediate Computer Programming

Class Section # 6

Jesse Farek

2/17/2015

**Analysis and Conclusions**

The point of this lab was to create different sorting methods. We found that the len function was very important in both of the sorts. We then used temporary variabels that were constantly rewritten to be able to switch the values without losing any of them. The selection sort was the hardest and required some help from online to figure it out, we were missing the 0,1,-1 parts in are code although im not entirely sure why

Computer was plugged in at 100%

Mac – OSX operating system

Ram- 4GB DDR3

Analysis Questions

1. The complexities of the two sort functions were very difficult, and depending on the length of the numbers you are computing can change the results on which one is best. This has a direct result and how long it takes and which one is faster.
2. There were only 2, the pros and cons depend on the length of your list. It can also depend on what you are sorting, the con of bubble is when your list is very long you are constantly going through the same length of numbers over and over, while for sort you take a number out and just read what is left, making each trip smaller and smaller.
3. Python sort method it was selected for us.
4. You could use a reverse sort, instead of doing from smallest to largest, you could do largest to smallest, this could be helpful in grades. Where you want largest as your first grades to appear meaning they did the best.
5. The length of the list changes how long the sorting will take and can change which one is the most efficient.

Source Code:

Corey Henry & natalie morrision #Date Assigned: 24Feb14

# #

#Course CSE 1384 Sec 06 #Date Due: 03Mar14

#File name: Lab

#

#Program description-

import random

import time

##function that will take the two numbers in a list side by side and determine which is smaller

#then will put the smallest infront

def bubbleSort(aList):

n = len(aList)

for each in range(n):

index =0

for each in range(n-1):

# determines which number is smaller and puts it first, then moves up one index

if aList[index] > aList[index+1]:

var1 = aList[index]

var2 = aList[index+1]

aList[index] = var2

aList[index+1] = var1

index +=1

else:

index +=1

return

# function that will sort by checking the whole list for the lowest value

#and placing it first, then continuing on and on. help from interactivepython.org

def selectionSort(aList):

for each in range(len(aList)-1,0,-1):

mini = 0

for index in range(1,each+1):

if aList[index] > aList[mini]:

mini = index

var1 = aList[each]

aList[each] = aList[mini]

aList[mini] = var1

def main():

#Create a list of 1000 random numbers

myList = []

for i in range(10):

x = random.randint(1, 10)

myList.append(x)

#Sort using python sort method

start = time.time()

myList.sort()

stop = time.time()

total\_time = stop - start

print("Sorted:", myList)

print("Time needed for python sort method: ", total\_time)

#Recreate an unordered list

for i in range(len(myList)):

myList[i] = random.randint(1, 10)

#Sort using bubble sort method

start = time.time()

bubbleSort(myList)

stop = time.time()

total\_time = stop - start

print("Sorted:", myList)

print("Time needed for bubble sort function: ", total\_time)

#Recreate an unordered list

for i in range(len(myList)):

myList[i] = random.randint(1, 10)

#Sort using selection sort method

start = time.time()

selectionSort(myList)

stop = time.time()

total\_time = stop - start

print("Sorted:", myList)

print("Time needed for selection sort function: ", total\_time)

input("\nPress Enter to close")

main()

