Michael Dubé

Dr. Dae Wook Kim

CSC 310

26 October 2018

**radixSort:**

In radix sort, there are two groups of nested for-loops (one outside and one inside), one to sort the last digit in each element in the set, and one to sort the first digit in each element in the set, and two for-loops, one to refill the main queue, and one to print off each element in the queue.

def radixSort(self):

tempQueue = ArrayQueue()

for i in range(0, 10):

for j in range (0, self.\_\_len\_\_()):

if(self.first()%10 == i):

tempQueue.enqueue(self.dequeue())

else:

self.enqueue(self.dequeue())

for t in range(0, tempQueue.\_\_len\_\_()):

self.enqueue(tempQueue.dequeue());

for i in range(0, 10):

for j in range (0, self.\_\_len\_\_()):

if(self.first()//10 == i):

tempQueue.enqueue(self.dequeue())

else:

self.enqueue(self.dequeue())

for i in range (0, tempQueue.\_\_len\_\_()):

print(tempQueue.dequeue())

For each number 0-9,

check the second digit of each element in queue

If the element matches that of the index, dequeuer the element and put it in the temporary queue

Otherwise, dequeue it and put it back into the original queue

For each element in the temporary queue,

dequeue the element and put it back into the original queue

For each number 0-9,

check the first digit of each element in queue

If the element matches that of the index, dequeuer the element and put it in the temporary queue

Otherwise, dequeue it and put it back into the original queue

For each element in the queue,

Dequeue the element and print it.

This results in a big-Oh notation of O(n2).

**fromPostFix:**

in the function evaluating an expression from postfix notation, there is one while loop, moving through the string input by the user and/or data files. Each time the while loop runs, it cuts off the beginning of the string to move on to the next character, and evaluates the algorithm’s next action based on what character it receives.

def fromPostFix(self, s):

Stak = ArrayStack()

while (len(s) > 0):

if (s[0] >= '1' and s[0] <= '9'):

Stak.push(int(s[0]))

elif (s[0] == '\*'):

Stak.push(Stak.pop()\*Stak.pop())

elif (s[0] == '/'):

t = Stak.pop()

Stak.push(Stak.pop()/t)

elif (s[0] == '-'):

t = Stak.pop()

Stak.push(Stak.pop()-t)

elif(s[0] == '+'):

Stak.push(Stak.pop()+Stak.pop())

else:

print("Incorrectly Entered")

s = s[1:len(s)]

print(Stak.pop())

While the input string is of length greater than 0,

If the first character is a number 0-9, put the number value into the stack

If it is an operation, pop the first two numbers off from the stack and evaluate them with the given operator

Cut off the first character of the string

Pop the last remaining numerical value from the stack and print it.

This results in a big-Oh notation of O(n).

**Works Cited**

array\_stacks.py and array\_queue.py from Dr. Dae Wook Kim’s lecture notes on Blackboard