Fundamentals Worksheet

# Review Questions

What does it mean for an algorithm to have O(1)/constant runtime?

As input scale increases, the runtime of the program doesn’t change.

Give an example or write a function with the following runtimes and explain how you know they have that runtime.

O(1) runtime:

def constant\_time(N):

for i in range(10):

print(i)

O(2^N) runtime:

def exponential\_ time(N):

if(N < 0):

return

print(N)

exponential\_time(N-1)

exponential\_time(N-1)

O(N\* lgN\* lgN) runtime:

Def wired\_ time(N):

for i in range(N):

for j in range(N):

for k in range(N):

k = 2\*k

j = 2\*j

i++

Go through the following functions line by line and label the runtime and space complexity for each line. Then figure out the overall runtime and space complexities.

#input: an arbitrarily large array of integers

function1(arr):

arr.sort() #N\*log(N)

reverseOfArr = [arr[i] for i in range(len(arr)-1, -1, -1)] #N

Compliments = []

#zip just iterates through both arrays at the same time

# zip operation N

For num1, num2 in zip(arr, reverseOfArr): # N\*1

Compliments .append(num1 + num2)

Return Compliments #N\*log(N) + N +N + N = 3\*N + N\*long(N) =(3 + log(N))\*N

if N>> 3 ->log(N) >>3, we have N\*log(N)

#input: an arbitrarily large integer

function2(N):

helperFunction()

For i in range(N): #N

helperFunction()

For j in range(N): #N\*log(N)

helperFunction()

helperFunction()

Return

#this function has lgN runtime

helperFunction()

#does something

O(N) -> N\*N\*log(N) ->N^2\*log(N)

# Give the order of growth (as a function of N) of the running times of each of the following code fragments:

**# Block (a)**

**sum = 0;**

**n = N**

**while n > 0: #N N/2… 0-> log(N)**

**for i in range(0, n): #(N)+(N/2)+(N/4)+… ->O(2N)->O(N)**

**sum += 1;**

**n = n // 2**

**Time Complexity = O(N)**

**# Block (b)**

**sum = 0**

**i = 1**

**while i < N: # 1 2 4… N->log(N)**

**for j in range(0, i): # 1+2+4+…+N ->O(2N)->O(N)**

**sum += 1**

**i = i \* 2**

**# Block (c)**

**sum = 0**

**i = 1**

**while i < N: #1 2 4… N ->log(N)**

**for j in range(0, N): #N+N+N+…+N->N\*log(N)**

**sum += 1**

**i = i \* 2**

**Question 1 -**

I have a stack and perform the following operations:

2-1, 1, 4-3-1, 3-1

stack.push(1)

stack.push(2)

stack.pop()

stack.push(3)

stack.push(4)

stack.pop()

stack.pop() = 3

What is the output of the last operation?

I have a queue and perform the following operations

1-2, 2, 2-3-4, 3-4, 4

queue.enqueue(1)

queue.enqueue(2)

queue.dequeue()

queue.enqueue(3)

queue.enqueue(4)

queue.dequeue()

queue.dequeue() = 3

What is the output of the last operation?

Which of these is a real world example of a stack? Choose all that apply

A. Dogpile in Football #heap

B. Pancakes #stack

C. Back button in Web Browser #stack

D. Line at the bank #queue

class LinkedList:

def \_\_init\_\_(self, listNode):

//head references first node in chain of nodes

self.head = listNode

def function1(self, new\_node):

new\_node.next = self.head

self.head = new\_node

def function2(self, newNode):

if self.head is None:

self.head = new\_node

return

cur = self.head

while(cur.next):

cur = cur.next

cur.next = newNode

def function3(self):

if self.head:

self.head = self.head.next

We have these 4 functions names that more accurately describe function1, function2 and function3

A) insertEnd

B) deleteFront

C) deleteEnd

D) insertBeginning

Question 4 - What name most accurately describes function1? D

Question 5 - What name most accurately describes function2? A

Question 6 - What name most accurately describes function3? B

Write some simple test cases (can just use a main method) for the above 3 functions to verify that your answers are correct.

Download the file [here](https://www.dropbox.com/s/dfnaqkew8tqz5zt/linkedListTestCases.py?dl=0)

# Tips for Solving Leetcode Problems

* Spend only 45 minutes trying to devise a solution to each problem (everything except coding)
* Please reference our [guide on how to approach practice exercises in Teachable](https://codebreakers1.teachable.com/courses/codebreakers-training-vault/lectures/14591190).

Complete the following Leetcode Problems and add them to your LC Review Schedule.

|  |
| --- |
| 1. [Intersection of Two Linked Lists](https://leetcode.com/problems/intersection-of-two-linked-lists/) |
| 1. [Remove Kth Node From End](https://leetcode.com/problems/remove-nth-node-from-end-of-list/) |
| 1. [Palindrome Linked List](https://leetcode.com/problems/palindrome-linked-list/) |
| 1. [Evaluate Reverse Polish Notation](https://leetcode.com/problems/evaluate-reverse-polish-notation/) |
| 1. [Number of Recent Calls](https://leetcode.com/problems/number-of-recent-calls/) |