CSCI 3104 Summer 2018
Problem Set 1
Eischen, Parker
10/31
parker.eischen@colorado.edu

1. (20 points) Give an example of an application that uses a proprietary algorithm (i.e. Spotify's "Discover Weekly" playlist, Google's PageRank algorithm, etc.). Find an article that discusses this algorithm and give a summary of its content.

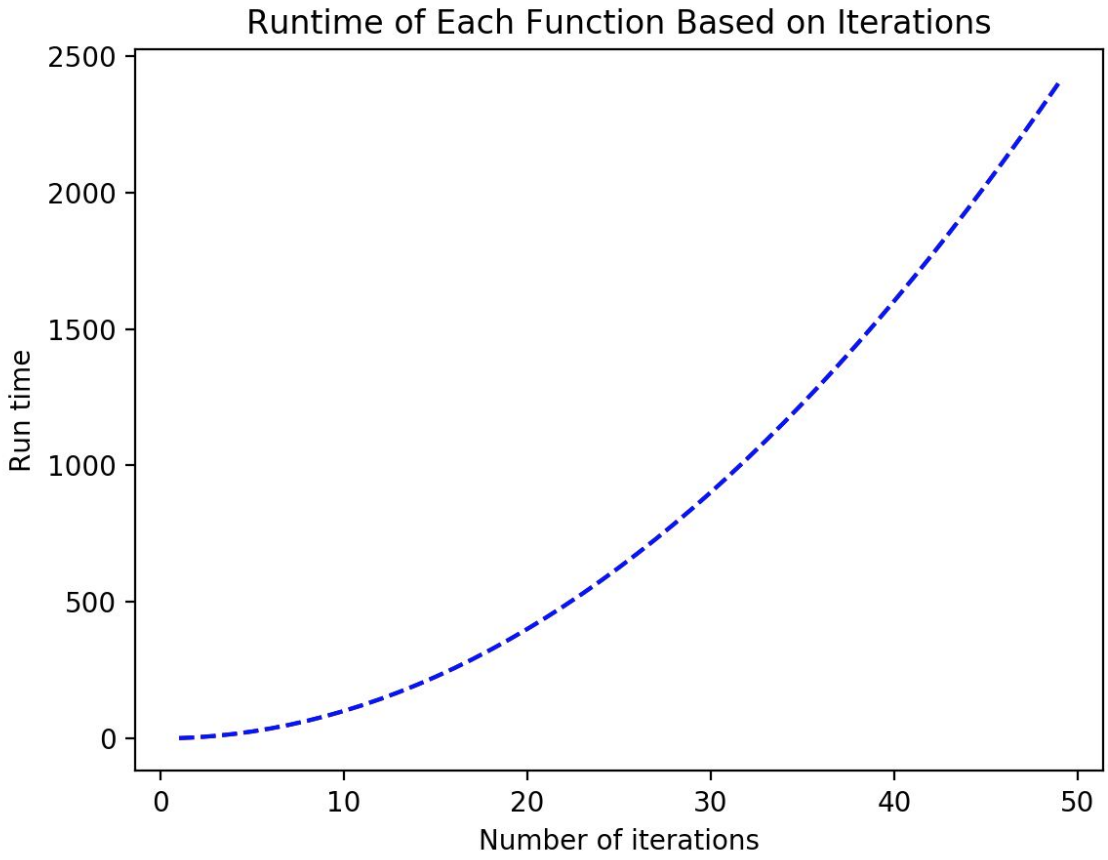
An application that uses a proprietary algorithm is Amazon.com's A9 algorithm. This algorithm is in charge of Amazon's search engine, making sure the products displayed are relevant to the user. The article *Amazon A9 Optimization & Algorithm* goes into more detail on the topic. Based on the article, the products that are shown are based off of relevancy + popularity metrics which is similar to Google's Page Rank. According to the Article, "70% of Amazon Customers never click past the first page of search results, 35% of Amazon shoppers click on the first product featured on a search page". The importance of the A9 is to make sure the customer makes a purchase while on the website. Some factors that affect the algorithm include product title, description,price, availability, and your sales history to name a few.

Articles used:

https://amazonseoconsultant.com/amazon-a9-optimization/
https://www.slideshare.net/CPCStrategy-ConvertRetailIntent/amazons-a9-organic-product-ranking-algorithm-unpacked

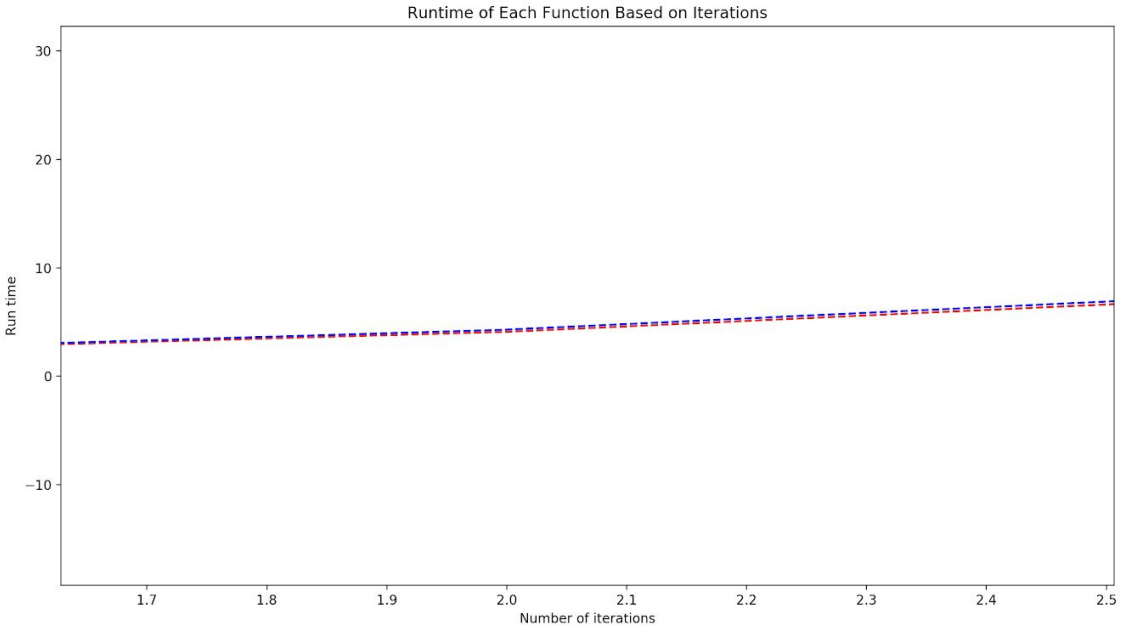
2. (50 points) Consider two algorithms that perform the same function, that run in n 2 + n/20 and n 2 + ln(n), respectively, where n ∈ N (i.e. natural numbers).

(a) Plot these runtimes on the same graph with the values n ∈ [1, 50] (don't forget labels). Provide the set of intervals over N, where n 2+n/20 is the better algorithm to use.



Red = n^2 + n/20
Blue = n^2 + ln(n)

It may look like there is only one function present, but both are indeed showing. The functions are so similar from one another that they are nearly identical.



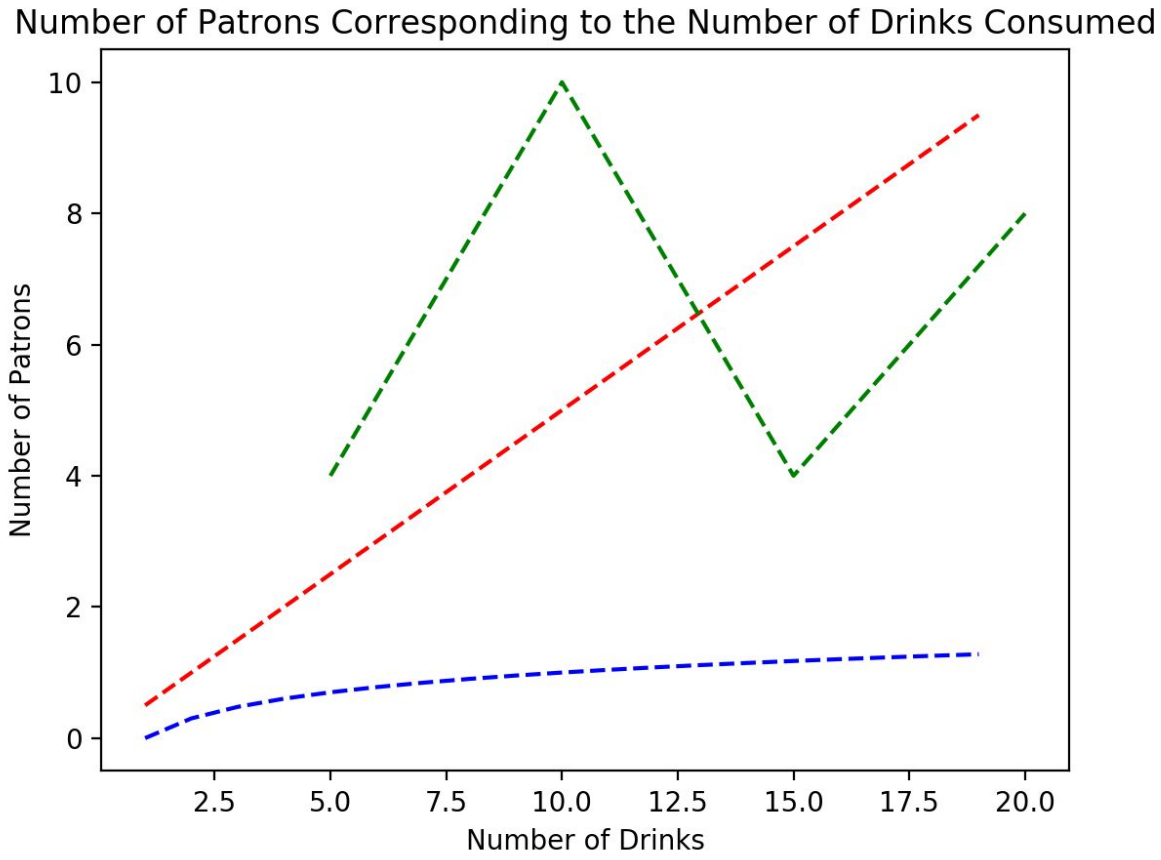
If you zoom into the graph you can see that there is indeed some overlap, and a crossing point from one function to another. The red graph is n^2 + n/20. The interval in which this graph is optimal is [2,50]. The function actually becomes more optimal when n = 1.0541. However, since we are dealing with natural numbers, n has to be an integer and we round up to n = 2.

(b) Harry the Wizard needs your help solving a riddle deep in an abandoned dwarven mine. There are two doors marked A and B, respectively, and a stone pedestal in the middle of the room inscribed with the following text:
"The dwarves who dwelled in this mine were fond of mathematical drinking games. Two dwarves Arnold and Barry are chosen as the participants for this game, and pick functions that they think will best predict the number of people who enter the pub in an hour (p) to the number of drinks they consume in that hour (d). They choose p(d) = d/2 and p(d) = ln(d), respectively. Below is a record of the number of drinks consumed and the corresponding number of patrons patrons who entered the bar over four hours.

Number Drinks	*Number Patrons*
5	*4*
10	*10*
15	*4*
20	*8*

Due to an error in Harry's mental calculations in the last puzzle causing Grog the Barbarian to lose his pinky finger, the party demands a written explanation of the solution to the puzzle. Additionally, since Grog doesn't know how to read, provide a relevant figure in your solution so Grog can believe he is part of the discussion.

Red = Arnold's Formula (d/2)
Blue = Barry's Formula (ln(d))
Green = Actual Results



Although neither of the dwarves guesses were identical to the actual results, the trend of the results is closer to Arnold's prediction(d/2). Barry's guess is way too low for the number of patrons expected per drink.

3. (30 points) Consider the following recurrence relation:

$$Gn = \begin{cases} 1 & \text{if } n = 0 \\ -1 & \text{if } n = 1 \\ 2 & \text{if } n = 2 \\ (Gn-1)(Gn-2)+Gn-3 & \text{otherwise} \end{cases}$$

(a) Write pseudocode for this function that takes in a positive integer, n, and returns the nth number in the sequence.

If recursion was used this function will take forever to run and the Time Complexity will be obnoxious.

```
def Recursion(n):
    if n == 1:
        return 1
    if n == 2:
        return -1
    if n == 3:
        return 2

    else:
        return (Recursion(n-1)* Recursion(n-2)+Recursion(n-3))
```

This algorithm has a
Space Complexity = O(n)
Time Complexity =
O(1) for basecases
O(3^n) //3 calls to the function for every single n except the base cases

```
def Function(n):
    A = [1,-1,2] #initializing the base cases
    for i in range(3,n):
        val = (A[i-1]*A[i-2])+A[i-3]
        A.append(val)
    return A[n-1]
```

This algorithm is way more efficient. Instead of using recursion to compute values that are already known, we store those values into an array to be used later. This gives the additional benefit of knowing all the values of the function, rather than just the wanted result.

This algorithm has a
Space Complexity = O(n)
Time Complexity =
O(1) //for basecases
O(n) //only one for loop, so the longest it will take is n iterations

(b) What is the 10th number in the sequence?

The 10th number in the sequence is **-110655.** This was derived by using a python function of the above code and using 10 as the n input. To confirm, I also tested this in the recursive function as well and got the same results.

Source Code:

Graph of P2a:

```
import matplotlib.pyplot as plt
import math

y=[]
z = []

for n in range(1,50):
        x = n**2 + n/20
        x2 = n**2 + math.log10(n)
        y.append(x)
        z.append(x2)

i = range(1,50)

plt.plot(i,y,'r--' )
plt.plot(i,z, 'b--')

plt.ylabel('Run time')
plt.xlabel('Number of iterations')
plt.title('Runtime of Each Function Based on Iterations')

plt.show()
```

Graph of P2b:

```
import matplotlib.pyplot as plt
import math

patrons = 0

Arnold = []
Barry = []

for d in range(1,20): #array for Arnolds guess
        patrons = d/2
        Arnold.append(patrons)

patrons =
for i in range(1,20): #array of Barrys guess
        patrons = math.log10(i)
        Barry.append(patrons)

ResultsX = [5,10,15,20]
ResultsY = [4,10,4,8]

x = range(1,20)

plt.plot(x,Arnold,'r--')
plt.plot(x,Barry,'b--')
plt.plot(ResultsX,ResultsY,'g--')

plt.ylabel('Number of Patrons')
plt.xlabel('Number of Drinks')
plt.title('Number of Patrons Corresponding to the Number of Drinks Consumed')

plt.show()
```