Relatório Projeto 1 AED 2023-2024

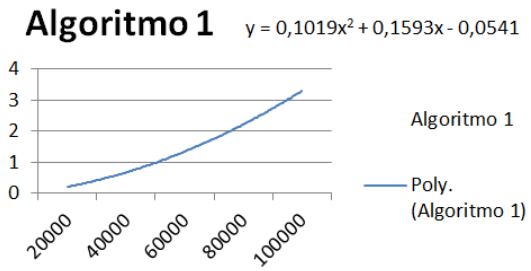
Nome: Rodrigo Borges Nº Estudante: 2022244993
PL (inscrição): 7

Registar os tempos computacionais das 3 soluções. Os tamanhos das arrays (N) devem ser: 20000, 40000, 60000, 80000, 100000. Só deve ser contabilizado o tempo do algoritmo. Exclui-se o tempo de leitura do input e de impressão dos resultados. Devem apresentar e discutir as regressões para as 3 soluções, incluindo também o coeficiente de determinação/regressão (r quadrado).

Tabela para as 3 soluções

Tamanho da entrada	Algoritmo 1	Algoritmo 2	Algoritmo 3
20000	0,21795104	0,00030005	0,0001111
40000	0,65066547	0,00051537	0,00040035
60000	1,3411778	0,0005969	0,00070376
80000	2,23573394	0,00091145	0,00082369
100000	3,27995548	0,00107529	0,00101974

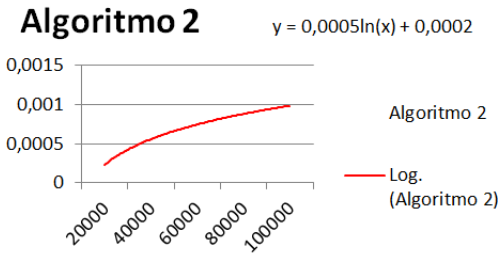
Gráfico para a solução A



Tipo de Regressão: Quadrática

Complexidade: O(n^2)

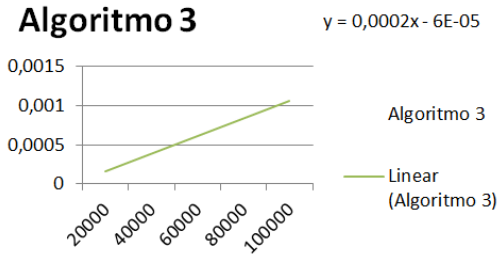
Gráfico para a solução B



Tipo de Regressão: Logarítmica

Complexidade: O(n(log(n))

Gráfico para a solução C



Tipo de Regressão: Linear

Complexidade: O(n)

CÓDIGO

```
import random
import time
import pandas as pd
def criar_input(n,k):
    lista=[]
    for i in range(0,n):
      a=random.randint(1,k-1)
      lista.append(a)
    return lista
def algoritmo_1(lista,k,n):
   booleano=False
   for i in range(0,n):
      for j  in range(i+1,n):
         if lista[i]+lista[j]==k and lista[i]!=lista[j]:
            booleano=True
            break
   return booleano
def algoritmo_2(lista,k,n):
   lista.sort()
   i=0
   j=n-1
   while i<j:
     if lista[i]+lista[j] == k and lista[i]!=lista[j]:
        return True
     elif lista[i]+lista[j]<k and lista[i]!=lista[j] :
        i+=1
     elif lista[i]+lista[j]>k and lista[i]!=lista[j]:
        j -=1
   return False
```

```
def algoritmo_3(lista, k, n):
    conjunto = set(lista)
    for i in lista:
        valor = k - i
        if valor in conjunto:
            return True
    return False
def calcula_tempo(k):
    tamanhos = [20000, 40000, 60000, 80000, 100000]
    resultados = {'Tamanho da entrada': [],'Algoritmo 1': [],
'Algoritmo 2': [], 'Algoritmo 3': []}
    for tamanho in tamanhos:
        lista = criar_input(tamanho, k)
        tempos_1 = []
        tempos_2 = []
        tempos_3 = []
        for i in range(10):
            tempo_inicial_1 = time.time()
            algoritmo_1(lista, k, tamanho)
            tempo_final_1 = time.time()
            tempos_1.append(tempo_final_1 - tempo_inicial_1)
        for j in range(30):
            tempo_inicial_2 = time.time()
            algoritmo_2(lista, k, tamanho)
            tempo_final_2 = time.time()
            tempos_2.append(tempo_final_2 - tempo_inicial_2)


            tempo_inicial_3 = time.time()
            algoritmo_3(lista, k, tamanho)
            tempo_final_3 = time.time()
```

```
            tempos_3.append(tempo_final_3 - tempo_inicial_3)

        resultados['Tamanho da entrada'].append(tamanho)

        resultados['Algoritmo 1'].append(sum(tempos_1) /
len(tempos_1))

        resultados['Algoritmo 2'].append(sum(tempos_2) /
len(tempos_2))

        resultados['Algoritmo 3'].append(sum(tempos_3) /
len(tempos_3))


    return resultados
```

Análise dos resultados tendo em conta as regressões obtidas e como estas se comparam com as complexidades teóricas:

O algoritmo 1 ao utilizar 2 for's aninhados torna a sua complexidade O(n^2), o algoritmo 2 ao utilizar o ordenamento faz com que a sua complexidade seja O(nlog(n), já o algoritmo 3 é o mais simples pois cria um conjunto a partir de uma lista(set) fazendo com que a sua complexidade seja O(n). Tendo em conta as regressões utilizadas e obtidas e as complexidades teóricas previamente explicadas, posso concluir que o resultado foi dentro daquilo que esperava. Em primeiro lugar a previsão da rapidez foi parecida visto que segundo a teoria o algoritmo com complexo O(n^2) seria o mais demorado seguido do O(nlog(n)) e do O(n)(mesmo que estes últimos tenham tempos bastante semelhantes). Em segundo lugar, os gráficos estão bastante semelhantes àqueles que teoricamente deveriam ser caso fosse perfeito, algoritmo 1 - gráfico de uma quadrática, algoritmo 2 - Gráfico de um Logaritmo e algoritmo 3 - Gráfico Linear