**MAC122 – Princípios de Desenvolvimento de Algoritmos**

**Segundo Semestre de 2021 – BMAC – IMEUSP – Prof. Marcilio**

**Exercício Programa - Entregar até 21/11/2021**

Dado um arquivo de texto, onde cada linha contém os seguintes campos separados por vírgula:

<identificação>,<nome>,<data nascimento>

Exemplo:

**32343245100,Jose de Castro,10/12/2001**

**43456790236,Maria das Dores e Silva,05/01/2005**

**22589349476,Antonio Paixao dos Santos,30/05/2012**

**32343245111,Jose de Castro,10/12/2010**

**43456790222,Maria da Silva,25/01/2015**

**22589348876,Antonio dos Santos,30/06/2002**

**32343245144,Jose de Campos,15/15/2007**

**43453330236,Maria da Rocha,27/11/2000**

**22555549476,Armando da Silva,20/06/2002**

**...**

O programa deve ler o arquivo, colocá-lo numa lista **TAB,** onde cada elemento é uma lista com 3 strings contendo cada um dos campos, classificar esta lista em ordem crescente de várias formas e comparar o tempo gasto em cada uma das classificações.

**TAB:**

**[["32343245100","Jose de Castro","10/12/2001"]**

 **["43456790236","Maria das Dores e Silva","05/01/2005"]**

 **["22589349476","Antonio Paixao dos Santos","30/05/2012"]**

 **. . .**

**]**

**A ordem de classificação**

Como são 3 campos, vamos numerá-los como campos 1 2 3

1 - <identificação>

2 - <nome>

3 - <data nascimento>

A ordem é sempre crescente e é dada por uma permutação de 1 2 3. Exemplos:

213 – Pelo <nome>, dentro deste pela <identificação>, dentro desta pela <data nascimento>

132 – Pela <identificação>, dentro desta pela <data nascimento>, dentro desta pelo <nome>

321 – Pela <data nascimento>, dentro desta pelo <nome>, dentro deste pela <identificação>

**Algoritmo de Classificação**

Escreva 3 funções:

1. ClassificaBolha(TAB, ordem) – que classifica TAB pelo método da Bolha
2. ClassificaQuick(TAB, ordem) – que classifica TAB pelo método Quick **não recursivo**
3. ClassificaSort(TAB, ordem) – que classifica TAB usando o método sort() do Python

O parâmetro ordem é uma permutação de 123.

**O programa**

O programa terá então a seguinte estrutura:

while True:

 Pedir o nome do arquivo de origem (‘fim’: break)

Ler o arquivo de origem e colocar em TAB (já com split(‘,’) dos campos)

while True:

 TAB1 = TAB2 = TAB3 = TAB (cópias de TAB)

Pedir a ordem (número inteiro que seja permutação de 123) (‘fim’: break

ClassificarSort(TAB1, ordem) – cronometrar e mostrar o tempo gasto

ClassificarBolha(TAB2, ordem) – cronometrar e mostrar o tempo gasto

Comparar TAB1 com TAB2 para ver se classificou corretamente

ClassificarQuick(TAB3, ordem) – cronometrar e mostrar o tempo gasto

Comparar TAB1 com TAB3 para ver se classificou corretamente

**Exemplo de saída do programa:**

**Nome do arquivo de origem:arq1.txt**

**Ordem da classificação:231**

**Tempo de classificação Sort() = 0.0130875205993652**

**Tempo de classificação Bolha = 1.0030875205993652**

**\*\*\* Classificação correta**

**Tempo de classificação Quick = 0.5030875205993652**

**\*\*\* Classificação correta**

**Ordem da classificação:132**

**Tempo de classificação Sort() = 0.0233333335993652**

**Tempo de classificação Bolha = 0.9923232332293652**

**\*\*\* Classificação correta**

**Tempo de classificação Quick = 0.3232323232993652**

**\*\*\* Classificação correta**

**Ordem da classificação:fim**

**Nome do arquivo de origem:arq2.txt**

**...**

**...**

**...**

**Nome do arquivo de origem:fim**

**O programa gerador dos arquivos de testes**

Use o programa abaixo para gerar arquivos de teste para o seu programa.

Gere por exemplo arquivos com 100, 200 ou 1000 registros usando o programa abaixo.

**from random import seed, randrange**

**# nomes randômicos**

**n1 = ["Felicia", "Catulo", "Osmund", "Artmio", "Senizio", "Tilenio"]**

**n2 = ["Cartuxo", "Olambro", "Romulo", "Ambulo", "Atomon", "Virino"]**

**n3 = ["Sereno", "Soterno", "Moncoes", "Oscaran", "Topovi", "Talento"]**

**n4 = ["Lasmia", "Mantega", "Casas", "Lorentao", "Melkioz", "Motivio"]**

**nn = 6**

**# Gera um registro com IDENT,NOME,DATAN**

**# Conteúdo randômico baseado em seu NUSP**

**# pp = '' - gera um registro completo**

**# pp != '' - gera apenas uma nova datan + ident ou apenas ident**

**def GeraRegistro(pp):**

 **global n1, n2, n3, n4, nn**

 **# nome, datan e ident**

 **nome = n1[randrange(nn)] + ' ' + n2[randrange(nn)] + ' ' + n3[randrange(nn)] + ' ' + n4[randrange(nn)]**

 **dia = randrange(28) + 1**

 **mes = randrange(12) + 1**

 **ano = randrange(17) + 2000**

 **datan = f'{dia:02}' + '/' + f'{mes:02}' + '/' + f'{ano:04}'**

 **ident = f'{randrange(100000000000):011}'**

 **kr = randrange(3)**

 **if pp == '':**

 **# gera um novo registro completo**

 **registro = ident + ',' + nome + ',' + datan**

 **return registro**

 **elif kr == 0:**

 **# preserva o nome e gera datan + ident**

 **campos = pp.split(',')**

 **registro = ident + ',' + campos[1] + ',' + datan**

 **return registro**

 **elif kr == 1:**

 **# preserva o nome e datan e gera ident**

 **campos = pp.split(',')**

 **registro = ident + ',' + campos[1] + ',' + campos[2]**

 **return registro**

 **else:**

 **# preserva ident e nome e gera datan**

 **campos = pp.split(',')**

 **registro = campos[0] + ',' + campos[1] + ',' + datan**

 **return registro**

**# gera arquivo nomearq com nreg registros**

**def GeraArquivo(nusp, nomearq, nreg):**

 **# randomize**

 **seed(nusp)**

 **# quantidade de registros - gera 80% do total**

 **nreg80 = nreg \* 80 // 100**

 **# tabela para guardar registros para repetição**

 **tab = ['' for k in range(nreg // 20)] # 5% dos registros**

 **# abre arquivo para gravação**

 **arq = open(nomearq, "w")**

 **# grava metade dos registros**

 **for k in range(nreg80):**

 **reg = GeraRegistro('')**

 **arq.write(reg + '\n')**

 **print(k + 1, " - ", reg)**

 **# guarda 5% dos registros para repetição**

 **if k % 20 == 0:**

 **# guarda em tab**

 **tab[k // 20] = reg**

 **# grava o resto dos 20% dos registros**

 **cont = nreg80 + 1**

 **for k in range(len(tab)):**

 **# para cada registro em tab gera 4 outros**

 **for j in range(4):**

 **reg = GeraRegistro(tab[k])**

 **arq.write(reg + '\n')**

 **print(cont, " - ", reg)**

 **cont += 1**

 **# fecha arquivo**

 **arq.close()**

**# Entre com seu NUSP - para randomizar**

**nusp = int(input("Entre com seu NUSP - para randomizar:"))**

**# Gera arquivo com uma certa quantidade de registros**

**while True:**

 **nome\_arq = input("Entre com o nome do arquivo.txt:")**

 **quant\_reg = int(input("Entre com a quantidade de registros:"))**

 **GeraArquivo(nusp, nome\_arq, quant\_reg)**