MAC122 – Princípios de Desenvolvimento de Algoritmos
Segundo Semestre de 2021 – BMAC – IMEUSP – Prof. Marcilio
Exercício Programa 3 - Entregar até 13/12/2021

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 objetivo do programa é listar todas as linhas do arquivo que tenham no campo <nome> alguma parte (nome ou sobrenomes) igual a um valor digitado.

Exemplo de saída do programa:

```
Entre com o nome do arquivo:meuarqr.txt

Entre com um valor:silva

43456790236,Maria das Dores e Silva,05/01/2005
43456790222,Maria da Silva,25/01/2015
22555549476,Armando da Silva,20/06/2002

* * * 5 comparações para localizar os nomes

Entre com um valor:MARia

43456790236,Maria das Dores e Silva,05/01/2005
43456790222,Maria da Silva,25/01/2015
43453330236,Maria da Rocha,27/11/2000

* * * 8 comparações para localizar os nomes
...

Entre com um valor:fim
```

Não considere como nomes as preposições (de, da, dos, etc.).
Para a comparação, considerar iguais as maiúsculas e minúsculas.
Não há letras acentuadas.

O programa deve ler o arquivo, colocá-lo numa lista **TAB,** onde cada elemento é uma linha do arquivo. Em seguida separar as partes dos nomes e criar uma tabela Hash com passo maior que 1 (hash duplo). Escolha uma boa função de hash (algum valor numérico sobre a string módulo o tamanho da tabela) e um tamanho de tabela conveniente.
Tem que colocar na tabela hash, além da parte do nome, a linha à qual pertence essa parte.

Exemplos:

TAB_HASH:

[["jose",0,3,6,...],...,["silva",1,4,8,...],..., ["paixao",2,...], ...]

O programa

O programa terá então a seguinte estrutura:

Ler o arquivo de entrada e colocar em TAB
Separar os nomes e construir a TAB_HASH
while True:
    Pedir uma parte do nome – se "fim": break
    Procurar e listar todas as linhas de TAB que contém essa parte do nome
    Mostrar quantas comparações com elementos de TAB_HASH foram feitas

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.
É o mesmo programa do EXPROG2.

```
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)
```