



**Escola Superior de Tecnologia e Gestão**

**Instituto Politécnico da Guarda**

AV. DR. FRANCISCO SÁ CARNEIRO, 50 – 6301-559 GUARDA TELF. 271 220 100

FAX. 271 222 690 GPS: Latitude:40.5416236730513,

Longitude: -7.28243350982666

**Algoritmos e estrutura de dados**

**Professor: Paulo Nunes**

Engenharia Informática: 2016/2017

Ano 1, 1º semestre.

# Gestão de passagens de automóveis em autoestradas

Trabalho de grupo

Daniel Ribeiro, Nº 1012527

Roberto Adelino, Nº 1012539



# Sumário:

- Apresentação do problema;
- Algoritmos;
- Estruturas de dados;
- Programa Python:
  - Estrutura: programa e menus;
  - Funcionalidades;
  - Demonstração (capturas de ecrã);
- Conclusão;



# Apresentação do problema

Através de um conjunto de dados fornecidos (( donos2.txt (10,000)),(carros.txt (10,000) )(cidades.txt (10))(distancias.txt (45))(passagens.txt (3,000,000) ) ) foi necessário o desenvolvimento de um programa em Python de modo a facilitar a organização, estrutura, consulta e tratamento desses mesmos dados.

## Gestão de carros

Inserir(txt)  
Inserir(bin)  
Alterar  
Eliminar  
Listar(txt)  
Listar (bin)  
Pesquisa por marca ou modelo  
Pesquisa por Marca ou modelo e Quantidade  
Converter carros.txt em carros.bin  
Voltar

## Gestão de cidades

Inserir  
Alterar  
Eliminar  
Listar todas  
Pesquisar por IDCidade  
Pesquisar por nome da cidade  
Voltar

## Gestão de distâncias

Inserir  
Alterar  
Eliminar  
Listar  
Pesquisa por cidade de partida  
Pesquisa por cidade de chegada  
Pesquisa distancia  
Pesquisar por preço  
Voltar

## Gestão donos

Inserir  
Alterar  
Eliminar  
Listar todos(txt)  
Listar todos(bin)  
Pesquisa por nome  
Pesquisa por Nome e Quantidade  
Converter donos.txt em donos.bin  
Voltar

## Menu principal

Donos  
Carros  
Cidades  
Distâncias (cidades)  
Passagens  
Terminar

## Gestão de passagens

Inserir  
Alterar  
Eliminar  
Listar todas  
Pesquisa por cidade (IDCidade)  
Pesquisa por IDUtenteFK  
Pesquisa por data  
Pesquisa por entrada ou saída  
Converter passagens.txt em passagens.bin  
Voltar

## Conclusão



# Menu principal: Algoritmo

## Início:

ESCREVER("1 - Dono")

ESCREVER("2 - Carros")

ESCREVER("3 - Cidades")

ESCREVER("4 - Distâncias")

ESCREVER("5 - Passagens")

ESCREVER()

ESCREVER("0 - Terminar")

ESCREVER()

Se (0) então

Fechar o programa

SeNão (1) então

gestaodedonos()

SeNão (2) então

gestaodecarros()

SeNão (3) então

gestaodecidades()

SeNão (4) então

gestaodedistancias()

SeNão (5) então

gestaodepassagens()

SeNão:

ESCREVER("Escolha uma opção contida na lista  
apresentada")

FimSe

FimSe

**Fim**



# Menu principal: Python

```
def menuprincipal():
```

```
    while True:
```

```
        print("1 - Dono")
        print("2 - Carros")
        print("3 - Cidades")
        print("4 - Distâncias")
        print("5 - Passagens")
        print()
        print("0 - Terminar")
        print()
```

```
        op=input("?")
        if op=="0":
            break
        elif op=="1":
            gestaodedonos()
        elif op=="2":
            gestaodecarros()
        elif op=="3":
            gestaodecidades()
        elif op=="4":
            gestaodedistancias()
        elif op=="5":
            gestaodepassagens()
        else:
            print("Escolha uma opção contida na lista  
apresentada")
```

```
    menuprincipal()
```



# Menu principal: Demonstração

```
C:\WINDOWS\py.exe

1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

? _
```

```
*Python 3.6.0 Shell*

File Edit Shell Debug Options Window Help
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:1
tel)] on win32
Type "copyright", "credits" or "license()" for more inf
>>>
= RESTART: F:\Nova pasta\Nova pasta\ (1012527)_WG\Trabal
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?
```





# Gestão donos : Algoritmo

## Início:

ESCREVER(1 – Inserir(txt)  
ESCREVER(2 – Alterar(txt)  
ESCREVER(3 – Eliminar(txt)  
ESCREVER(4 - Listar Todos(txt)  
ESCREVER(5 – Listar Todos(bin)  
ESCREVER(6 - Pesquisa por nome(txt)  
ESCREVER(7 - Pesquisa por Nome e Quantidade(txt)  
ESCREVER(8 – Converter donos txt em donos bin)  
ESCREVER()  
ESCREVER(0 - Voltar)

Se (0) então

voltar atrás

SeNão (1) então

inserirdono()

SeNão (2) então

ReadAllLinesVectorAlterarDono()

SeNão (3) então

ReadAllLinesVectorEliminarDono()

SeNão (4) então

Listatodosdonos()

SeNão (5) então

listarDonosbin()

SeNão (6) então

pesquisapornome()

SeNão (7) então

pesquisaporNomeEQuantidade()

SeNão (8) então

ConverteDonosTextoEmDonosBinario()

SeNão:

ESCREVER(Escolha uma opção  
contida na lista apresentada)

FimSe.

FimSe.

**Fim.**





# Gestão de donos: Python

```
def gestaoededonos ():  
    while True:  
        print("1 - Inserir dono (.txt)")  
        print("2 - Alterar dono (.txt)")  
        print("3 - Eliminar dono (.txt)")  
        print("4 - Listar todos os donos (.txt) ")  
        print("5 - Listar todos os donos (.bin) ")  
        print("6 - Pesquisa por Nome (.txt)")  
        print("7 - Pesquisa por Nome e Quantidade (.txt)")  
        print("8 - Converter donos2.txt em donos2.bin")  
        print()  
        print("0 - Voltar")  
        print()
```

```
op=input("?")  
    if op=="0":  
        break  
    elif op=="1":  
        inserirdono()  
    elif op=="2":  
        ReadAllLinesVectorAlterarDono()  
    elif op=="3":  
        ReadAllLinesVectorEliminarDono()  
    elif op=="4":  
        listatodosdonos()  
    elif op=="5":  
        listarDonosbin()  
    elif op=="6":  
        pesquisapornome()  
    elif op=="7":  
        pesquisaporNomeEQuantidade()  
    elif op=="8":  
        ConverteDonosTextoEmDonosBinario()  
    else:  
        print("Escolha uma opção contida na lista  
apresentada")
```



# Gestão de donos: Demonstração

```
C:\WINDOWS\py.exe

1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?1
1 - Inserir dono (.txt)
2 - Alterar dono (.txt)
3 - Eliminar dono (.txt)
4 - Listar todos os donos (.txt)
5 - Listar todos os donos (.bin)
6 - Pesquisa por Nome (.txt)
7 - Pesquisa por Nome e Quantidade (.txt)
8 - Converter donos2.txt em donos2.bin

0 - Voltar

?
```

```
*Python 3.6.0 Shell*

File Edit Shell Debug Options Window Help

Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:10) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\Roberto\Desktop\ALGORITMOS\1012527-1012539\WG\TrabalhoDeGrupo_1012527-1012539.py
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?1
1 - Inserir dono (.txt)
2 - Alterar dono (.txt)
3 - Eliminar dono (.txt)
4 - Listar todos os donos (.txt)
5 - Listar todos os donos (.bin)
6 - Pesquisa por Nome (.txt)
7 - Pesquisa por Nome e Quantidade (.txt)
8 - Converter donos2.txt em donos2.bin

0 - Voltar

?
```



# inserirdono(): Algoritmo

Algoritmo: Inserir\_Dono

Objetivo: Permite inserir um dono no ficheiro de texto donosNOVO.txt.

Constantes: FICHEIRO\_DONOSNOVO (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: donosNOVO.txt)

Variáveis:

Entrada:

IDUtente (TEXTO 9) - Dono do carro (>=1, <=999999999)

Nome (TEXTO 90) - Nome do dono do carro (Letras de A-Z)

NumeroDispositivoEletronico (TEXTO 9)- Numero do dispositivo eletrónico (<=20000,>1; - ;<1000,>=1;)

Saída:

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro  
donosNOVO.txt para  
escrita no fim

ESCREVER os dados no  
ficheiro

FECHAR o ficheiro  
Fim.



# inserirdono():Python

```
def inserirdono():  
    idutente = eval(input("IDUtente ?"))  
    nome = input("Nome ?")  
    nde = input("Numero Dispositivo eletrónico?")  
    ref_ficheiro = open("donosNOVO.txt", "at")  
    print(idutente, ";", nome, ";", nde, file=ref_ficheiro)  
    ref_ficheiro.close()
```



# inserirdono():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar
```

```
?1
IDUtente ?12345
Nome ?João
Numero Dispositivo eletrónico?1234
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar
```

```
?1
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar
```

```
?1
IDUtente ?12345
Nome ?João
Numero Dispositivo eletrónico?1234
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar
```

```
?|
```



# ReadAllLinesVectorAlterarDono():Algoritmo

Algoritmo: alterar\_donos

Objetivo: Permite alterar os dados de um dono á escolha.

Constantes: FICHEIRO\_DONOS2 (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: donos2.txt)

Variáveis:

Entrada:

IDUtente (TEXTO 9) - Dono do carro ( $\geq 1, \leq 999999999$ )

NumeroDispositivoEletronico (TEXTO 9) - Numero do dispositivo electrónico ( $\leq 20000, > 1$ ; - ;  $< 1000, \geq 1$ ;) )

Nome (TEXTO 90) - Nome do dono do carro (Letras de A-Z)

Saída:

Início:

Ler dados

ABRIR ficheiro donos2.txt

ESCREVER os dados no  
ficheiro donos2.txt

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# ReadAllLinesVectorAlterarDono(): Python (pt1)

```
def ReadAllLinesVectorAlterarDono():
```

```
    fname = 'donos2.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchIDUtente = input("IDUtente a alterar?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 3):
```

```
            continue
```

```
        IDUtente = colunas[0]
```

```
        Nome = colunas[1]
```

```
        NumeroDispositivoEletronico = colunas[2]
```

```
        if (SearchIDUtente == IDUtente):
```

```
            pos = i
```

```
            print ('\n' * 3)
```

```
            print ('IDUtente encontrado na posição ', i)
```

```
            print ('\n' * 2)
```

```
            print ('IDUtente.....:', IDUtente)
```

```
            print ('Nome.....:', Nome)
```

```
            print ('Numero Dispositivo Eletronico....:',
```

```
NumeroDispositivoEletronico)
```

```
            op = input("Alterar ? (s/n) ?")
```



# ReadAllLinesVectorAlterarDono(): Python (pt2)

```
if (op == 's'):
    print('Alterar ')
    IDUtente = input("IDUtente ?")
    Nome      = input("Nome ?")
    NumeroDispositivoEletronico = input("Numero Dispositivo Eletronico ?")
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
        else:
            print(IDUtente, Nome, NumeroDispositivoEletronico, sep=';', file=outfile)
    outfile.close();
    break;
```

if (pos == -1):  
 print('Esse Utente não existe')





# ReadAllLinesVectorAlterarDono():Demonstração

```
C:\WINDOWS\py.exe
?2
Registos: 19998
IDUtente a alterar?13290205

IDUtente encontrado na posição 934

IDUtente..... : 13290205
Nome..... : Carla Bacelar Real Azenha Peres
Numero Dispositivo Eletronico.... : 12894-116

Alterar ? (s/n) ?s
Alterar
IDUtente ?123456
Nome ?Maria
Numero Dispositivo Eletronico ?45342-123
```

```
?2
Registos: 19998
IDUtente a alterar?13290205

IDUtente encontrado na posição 10

IDUtente..... : 13290205
Nome..... : Carla Bacelar Real Azenha Peres
Numero Dispositivo Eletronico.... : 12894-116

Alterar ? (s/n) ?s
Alterar
IDUtente ?123456
Nome ?Maria
Numero Dispositivo Eletronico ?45342-123
```



# ReadAllLinesVectorEliminarDono():Algoritmo

Algoritmo: eliminar\_donos

Objetivo: Permite eliminar os dados de um dono à escolha.

Constantes: FICHEIRO\_DONOS2 (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: donos2.txt)

Variáveis:

Entrada:

IDUtente                      (TEXTO 9)      - Dono do carro                      ( $\geq 1, \leq 999999999$ )

Saída:

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

Ler dados

ABRIR ficheiro donos2.txt

ELIMINAR os dados do  
ficheiro donos2.txt  
referentes ao IDUtente  
pesquisado

FECHAR o ficheiro  
Fim.



# ReadAllLinesVectorEliminarDono():Python (pt1)

```
def ReadAllLinesVectorEliminarDono():  
    fname = 'donos2.txt'  
    infile = open(fname, "r")  
    lines = infile.readlines()  
    infile.close()  
    print("Registos: ", len(lines))  
    SearchIDUtente = input("IDUtente ?")  
    pos = -1  
    for i in range(len(lines)):  
        colunas = lines[i].split(';')  
        if (len(colunas) < 3):  
            continue  
        IDUtente = colunas[0]  
        Nome = colunas[1]  
        NumeroDispositivoEletronico = colunas[2]  
  
        if (SearchIDUtente == IDUtente):  
            pos = i  
            print ('\n' * 3)  
            print ('IDUtente encontrado na posição ', i)  
            print ('\n' * 2)  
            print ('IDUtente.....:', IDUtente)  
            print ('Nome.....:', Nome)  
            print ('Numero Dispositivo Eletronico....:', NumeroDispositivoEletronico)  
            op = input("Eliminar ? (s/n) ?")
```



# ReadAllLinesVectorEliminarDono():Python(pt2)

```
if (op == 's'):
    print('Eliminando ... ')
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
    outfile.close();
    break;
if (pos == -1):
    print('Esse Utente não existe')
```



# ReadAllLinesVectorEliminarDono():Demonstração

```
?3
Registos: 19998
IDUtente ?123456
```

```
IDUtente encontrado na posição 10
```

```
IDUtente..... : 123456
Nome..... : Maria
Numero Dispositivo Eletronico.... : 45342-123
```

```
Eliminar ? (s/n) ?s
```

```
Eliminando ...
```

- 1 - Inserir
- 2 - Alterar
- 3 - Eliminar
- 4 - Listar Todos
- 5 - Pesquisa por Nome
- 6 - Pesquisa por Nome e Quantidade
- 0 - Voltar

```
?3
Registos: 19997
IDUtente ?123456
```

```
IDUtente encontrado na posição 933
```

```
IDUtente..... : 123456
Nome..... : Maria
Numero Dispositivo Eletronico.... : 45342-123
```

```
Eliminar ? (s/n) ?s
```

```
Eliminando ...
```

- 1 - Inserir
- 2 - Alterar
- 3 - Eliminar
- 4 - Listar Todos
- 5 - Pesquisa por Nome
- 6 - Pesquisa por Nome e Quantidade
- 0 - Voltar

```
?|
```



# listatodosdonos():Algoritmo

Algoritmo: listar\_todos\_donos

Objetivo: Permite listar o nome de todos os donos.

Constantes: FICHEIRO\_DONOS2 (TEXTO 90) - Nome do ficheiro dos dados dos donos  
(Valor: donos2.txt)

Variáveis:

Entrada:

Saída:

IDUtente	(TEXTO 9)	- Dono do carro	(>=1, <=999999999)
Nome	(TEXTO 90-	Nome do dono do carro	(Letras de A-Z)
NumeroDispositivoEletronico	(TEXTO 9)	- Numero do dispositivo eletrónico	(<=20000,>1; - ; <1000,>=1;)

Início:

ABRIR ficheiro donos2.txt

ESCREVER os dados do  
ficheiro donos2.txt

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0



# listatodosdonos(): Python

```
def listatodosdonos():  
    fname = 'donos2.txt'  
    infile = open(fname, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDUtente= colunas[0]  
        Nome= colunas[1]  
        NumeroDispositivoEletronico= colunas[2]  
        print ("{0:15}{1:^60}{2:>15}".format(IDUtente, Nome, NumeroDispositivoEletronico))  
    infile.close()
```



# listatodosdonos():Demonstração

```
C:\WINDOWS\py.exe
```

5432782	Carlota Brito Andrade Rego Pinheiro Pacheco	6012-293
13084703	Luis Rosado Castanheira Coelho Carneiro Azevedo	2891-826
8841180	Liliana Azenha Regueira Mourão Eanes Mourão Caneco	16611-771
14656980	Ana Cabral Caiado Mateus Soeiro Fonseca	19879-792
1864455	Celina Domingos Moreira Almada Caiado	15897-180
8476864	Ricardo Proenca Brito Rego Peres Baptista	3376-71
1689322	Claudio Mora Santos Ramalho	8982-480
5072236	Rodolfo Rego Ramalho Rosado Castanheira	16489-184
10187115	Luis Pinheiro Soeiro Carrasco Calheiros	3238-298
1915194	Silvia Carmona Castanheira Estrela	17124-329
12537039	Joana Caneco Carmona Tinoco Rosado Silva	3075-850
7530569	Paula Mourão Bogas Calado	1312-114
7947918	Valdemar Seixas Almada Seixas Cipriano Bogas Real	8864-886
14027583	Pedro Paredes Ferreira Cipriano	2071-973
5371668	Bruna Moreno Belchior Carmona Pinto Clementino Queiros	12713-329
8186837	Celso Baptista Tinoco Rosado	6758-907
7040348	Bruna Quadros Proenca Nunes Cabral Brito	4086-719
7011614	Alzinda Mota Andrade Aguiar Ferreira	13934-761
2112193	Carla Rosado Leão Calado	2052-579
9808128	Silvia Farias Moura Ferro Brito	12724-271
5691013	Bruno Cipriano Paredes Matos Pinheiro	19781-495
7706644	Dora Coelho Baptista Carneiro Antunes Moreno	11826-458
5298637	Silvia Domingos Pires Andrade Estrela Dinis	10082-25
13731108	Pedro Soeiro Matos Carreira	4739-539
4565630	Celino Ferro Arantes Silva	5822-907
14621588	Cristina Azeredo Tinoco Sequeira Clementino Proenca Mateus	17399-116
8380376	Soraia Pinheiro Quadros Rosado	7649-526
1093319	Dulce Regueira Azeredo Eanes Sequeira Andrade Dinis	11067-270
2365412	Ricardo Cardoso Moreno Castanheira Nunes Almada Calheiros	4432-213
5496948	Odete Farias Rocha Mateus Rocha	1394-462

```
*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help

1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar

?4
IDUtente                               Nome                               Numer
oDispositivoEletronico
11319864                               Rita Mota Cipriano Mora Mendes Castanheira Domingos
9513-558
5460389                               Catarina Sequeira Mora Mora
17981-818
4699461                               Celino Soeiro Pinto Coelho Rego Cardoso
19018-520
5635454                               Valdemar Azeredo Carrico Pinheiro Estrela Mourão
11500-299
9605182                               Silvia Paredes Bacelar Arouca Arouca
19389-936
8896310                               Odete Cabral Estrela Cipriano
15066-794
14800521                               Basilio Amorim Rego Almeida Castro Farias Carreira
1547-622
4734952                               Basilio Real Pacheco Cabral Dias
14799-562
2437909                               Clarice Queiros Ramalho Seixas Matos
11224-945
11052900                               Soraia Macedo Mateus Carreira Moreira Azevedo Queiros
2965-146
3458750                               Clotilde Ramalho Calheiros Soeiro Carmona Soeiro
18671-976
11196045                               Carina Leal Ferraz Valbom Rosado Nogueira
4936-335
12759424                               Cristiana Cabral Brito Barros
11618-723
12041480                               Dora Ferraz Fonseca Nunes Martins Bacelar Nascimento
18721-930

Ln: 11372 Col: 35
```





# listarDonosbin(): Algoritmo

Algoritmo: listar\_todos\_donos

Objetivo: Permite listar o nome de todos os donos do ficheiro binário.

Constantes: FICHEIRO\_DONOS2 (binário) - Nome do ficheiro dos dados dos donos (Valor: donos2.bin)

Variáveis:

Entrada:

Saída:

IDUtente	(TEXTO 9)	- Dono do carro	(>=1, <=999999999)
Nome	(TEXTO 90-	Nome do dono do carro	(Letras de A-Z)
NumeroDispositivoEletronico	(TEXTO 9)	- Numero do dispositivo eletrónico	(<=20000,>1; - ; <1000,>=1;)

Início:  
ABRIR ficheiro donos2.bin

ESCREVER os dados do  
ficheiro donos2.bin

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0



# listarDonosbin():Python

```
def listarDonosbin():  
    import struct  
    donoFormato = struct.Struct('i80s20s');  
    f_bin = open("donos2.bin","rb")  
    f_bin.seek(0,2)  
    r = int(f_bin.tell() / donoFormato.size)  
    f_bin.seek(0,0)  
    for i in range (0,r):  
        donoBinario = f_bin.read(donoFormato.size)  
        IDUtente, Nome, NumeroDispositivoEletronico = donoFormato.unpack(donoBinario)  
        Nome = Nome.decode()  
        NumeroDispositivoEletronico = NumeroDispositivoEletronico.decode()  
        print("{0:<12}".format(IDUtente),end="")  
        print("{0:50}".format(Nome), end="")  
        print("{0:15}".format(NumeroDispositivoEletronico))  
    f_bin.close
```



# listarDonosbin():Demonstração

```
C:\WINDOWS\py.exe
10399615 Bóris Cardoso Matos Mendes Nunes Carneiro 13841-329
7966927 Celso Santos Martins Bacelar Quadros 4948-223
7001257 Ana Carrasco Martins Carmona Almeida Pinheiro Mateus 5218-401
3260445 Liliana Silva Cardoso Antunes Moreira Azevedo Paredes 9743-991
6077773 Bóris Leal Queiros Farias Amorim 15554-121
3205741 Bruna Matos Cabral Ferraz Tinoco 11734-722
4896940 Dora Moreno Carmona Seixas Barbosa 12823-78
7345463 Paula Amado Real Soeiro 11359-389
6474874 Dulce Regueira Mora Mendes Carneiro Queiros 3714-824
9216554 Claudio Peres Soeiro Nogueira Macedo Queiros 1459-525
6267724 Paula Eanes Rego Bogas 12854-883
14783014 Dina Domingos Calado Lobato 3097-718
10630597 Joana Macedo Martins Baptista Moreno Rego 9236-771
13594782 Rita Caiado Peres Santos 11303-644
3188651 Dolores Regueira Matos Azevedo Arantes Antunes 17238-793
14743720 Soraia Brito Carneiro Ramalho Amorim Lobato Moreira 15693-624
14801765 Rodolfo Rocha Moreno Leão Azeredo Amado 16868-617
12115955 Celino Azenha Regueira Nascimento Rosado Azenha 9970-805
4606082 Cristiana Dias Calado Rocha 18779-750
11866933 Claudio Nunes Clementino Brito Mourão 2960-814
3092141 Francisco Castanheira Amorim Peres Seixas 14926-122
3439460 Francisco Cardoso Estrela Carvalheira Azenha Carvalheira 1293-331
12847240 Dolores Rego Amado Andrade Castro Peres 8520-559
13976428 Clarice Seixas Brito Amorim Lobato Belchior 15382-744
14051098 Dulcineia Baptista Cabral Seixas 19232-81
14112593 Odete Martins Carvalheira Macedo Carmona 2382-105
9440255 Carlota Andrade Caneco Cardoso Rego 7122-628
3097089 Paula Castro Peres Santos Amora Bacelar Mora 8654-54
13923975 Hugo Ramalho Calheiros Nascimento Amado Amado Vivas 5667-209

*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help
14931-926
11280344 Rodolfo Calheiros Valbom Amorim Pires Leal Almada
4679-306
9618803 Hugo Mourão Carrasco Arouca Mendes
8245-79
11974722 Liliana Quadros Dias Domingos
19297-596
6591484 Ricardo Santos Lobato Tinoco Amorim
13576-654
6609229 Dolores Mora Ferraz Carrasco Seixas
16528-337
7513208 Rodolfo Ferro Eanes Almeida
16493-896
10328979 Ana Pires Ferraz Clementino Almeida
11974-362
12659028 Dolores Bogas Proenca Calheiros Carvalheira Vivas Andrade
7613-603
8803444 Francisca Mendes Estrela Seixas Belchior Calheiros Arantes
14172-206
13629929 Ricardo Rosado Paredes Peres
12328-53
9439316 Cristiana Carneiro Pacheco Ramalho Mora Martins
2867-816
1374224 Eduardo Paredes Carvalheira Real Proenca
7785-492
2721221 Basilio Mendes Cunha Amora Leão Carneiro
1203-914
12953361 Liliana Mota Ferreira Almeida Queiros
11824-725
4829064 Dolores Arouca Rego Eanes
16402-689
12426262 Joana Farias Eanes Amorim Quadros Barbosa Carvalheira
12164-571
7625134 Silvia Moreno Bacelar Clementino Martins Amora
1779-175
11615284 Catarina Nogueira Belchior Leal Quadros Antunes Leão
11210-958
10912290 Clotilde Leal Clementino Pacheco Rocha Cunha
9274-789
12722998
```



# pesquisapornome():Algoritmo

Algoritmo: pesquisa\_por\_nome

Objetivo: Permite pesquisar donos através do seu nome.

Constantes: FICHEIRO\_DONOS2 (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: donos2.txt)

Variáveis:

Entrada:

Nome (TEXTO 90) - Nome do dono do carro (Letras de A-Z)

Saída:

IDUtente (TEXTO 9) - Dono do carro ( $\geq 1$ ,  $\leq 999999999$ )

Nome (TEXTO 90) - Nome do dono do carro (Letras de A-Z)

NumeroDispositivoEletronico (TEXTO 9) - Numero do dispositivo eletrónico ( $\leq 20000, > 1$ ;  $< 1000, > 1$ ;

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro donos2.txt

ESCREVER IDUtente

ESCREVER Nome

ESCREVER

NumeroDispositivoEletronico

FECHAR o ficheiro

Fim.



# pesquisapornome():Python

```
def pesquisapornome():  
    nomeProcurar = input("Diga o nome a procurar?")  
    fname = 'donos2.txt'  
    infile = open(fname, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDUtente= colunas[0]  
        Nome= colunas[1]  
        NumeroDispositivoEletronico= colunas[2]  
        if (Nome.find(nomeProcurar)>=0):  
            print ("{0:15}{1:^60}{2:>15}".format(IDUtente,Nome,NumeroDispositivoEletronico))  
    infile.close()
```



# pesquisapornome(): Demonstração

```
C:\WINDOWS\py.exe
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade
0 - Voltar

?5
Diga o nome a procurar?Maria
5601627      Maria Pires Seixas Domingos Amora Arantes Mendes      16403-978
12300734     Maria Ferreira Cunha Cabral Regueira Calheiros            8467-675
1147987      Maria Nascimento Fonseca Real Carneiro Rocha Leal          18080-562
9628633      Maria Rocha Eanes Cipriano Lopes                            5833-600
10004418     Maria Pinheiro Proenca Pires Amorim Barros                  10034-629
4487529      Maria Castro Moreno Rocha                                    3969-163
4166300      Maria Mateus Bogas Real Brito Nunes                          2512-640
12957277     Maria Leão Moura Baptista Carneiro Matos Mourão             13951-588
4821932      Maria Almada Pinto Amorim Quadros Mora                      4884-555
8234083      Maria Arouca Bogas Cipriano Queiros                          10031-528
4360131      Maria Carmona Ramalho Amorim Arantes Carmona Cardoso        2416-534
2528401      Maria Amorim Ramalho Rosado                                   13418-951
1497367      Maria Farias Coelho Matos Pinheiro Azenha Cipriano           4325-554
5394548      Maria Valbom Nunes Antunes                                    4089-519
1124507      Maria Andrade Carreira Ferro Arantes Mora Amorim            2844-988
8584964      Maria Carreira Carneiro Peres Fonseca Moura Bogas           6308-176
4332984      Maria Barbosa Leão Pinheiro Caneco Nunes                     11194-865
13928858     Maria Brito Lopes Calado                                      12754-955
13871064     Maria Caiado Carvalheira Soeiro Almada Paredes               1999-21
```

```
*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade
0 - Voltar

?5
Diga o nome a procurar?Hugo
9618803      Hugo Mourão Carrasco Arouca Mendes
8245-79
13033353     Hugo Pires Carmona Fonseca
10863-393
5772917      Hugo Soeiro Moura Moura Farias Bogas
13875-16
12670728     Hugo Paredes Mourão Andrade Andrade Caiado Arouca
11017-580
4997045      Hugo Cunha Soeiro Dias Pinheiro
10456-891
12622831     Hugo Matos Aguiar Pires
2672-127
7940200      Hugo Carvalheira Carmona Castro Coelho Lopes Paredes
14100-669
9311931      Hugo Pires Pinto Rego
18944-655
6605321      Hugo Coelho Amado Lopes Vivas Barbosa
7184-328
13913674     Hugo Proenca Castro Eanes Rego Arantes
6432-111
12951118     Hugo Seixas Carrico Dinis
9561-928
9874674      Hugo Brito Domingos Regueira Mora Peres
3257-829
9489277      Hugo Leal Calheiros Lobato Azevedo Rocha
3252-750
9711511      Hugo Sequeira Matos Ferraz Andrade Paredes Macedo
10352-641
12986127     Hugo Bacelar Cipriano Belchior Castro

Ln: 20066 Col: 45
```



# pesquisaporNomeEQuantidade():Algoritmo

Algoritmo: pesquisa\_por\_nome\_e\_quantidade

Objetivo: Permite pesquisar quantos donos há com o nome que se pretende.

Constantes: FICHEIRO\_DONOS2 (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: donos2.txt)

Variáveis:

Entrada:

Nome (TEXTO 90) - Nome do dono do carro (Letras de A-Z)

Saída:

IDUtente (TEXTO 9) - Dono do carro ( $\geq 1$ ,  $\leq 999999999$ )

Nome (TEXTO 90) - Nome do dono do carro (Letras de A-Z)

NumeroDispositivoEletronico (TEXTO 9) - Numero do dispositivo eletrónico ( $\leq 20000, \geq 1$ ; - ;  $< 1000, \geq 1$ ;) ;

n (TEXTO 3) - Numero de pessoas com o mesmo nome ( $\geq 0, \leq 999$ )

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro donos2.txt

ESCREVER IDUtente

ESCREVER Nome

ESCREVER  
NumeroDispositivoEletronico

ESCREVER n (número de  
pessoas)

Fim.



# pesquisaporNomeEQuantidade():Python

```
def pesquisaporNomeEQuantidade():  
    nomeProcurar = input("Diga o nome a procurar?")  
    fname = 'donos2.txt'  
    infile = open(fname, "r")  
    n = 0  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDUtente= colunas[0]  
        Nome= colunas[1]  
        NumeroDispositivoEletronico= colunas[2]  
        primeiroNome = Nome.split(" ")  
        primeiroNome = primeiroNome[0]  
        if nomeProcurar==primeiroNome:  
            n = n + 1  
    print("Numero de pessoas com o nome",nomeProcurar, n)  
    infile.close()
```





# pesquisaporNomeEQuantidade():Demonstração

```
?6
Diga o nome a procurar?Luisa
Numero de pessoas com o nome Luisa 0
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar

?6
Diga o nome a procurar?Luis
Numero de pessoas com o nome Luis 388
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar

?
```

```
123456 111
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Nome
6 - Pesquisa por Nome e Quantidade

0 - Voltar

?6
Diga o nome a procurar?Maria
Numero de pessoas com o nome Maria 394
1 - Inserir
```



# ConverteDonosTextoEmDonosBinario():Python

```
import struct
```

```
donoFormato = struct.Struct('i80s20s');
```

```
f_texto = open('donos2.txt', "r")
```

```
f_bin = open("donos2.bin", "wb")
```

```
lines = f_texto.readlines();
```

```
for i in range(1, len(lines)):
```

```
    line = lines[i]
```

```
    line = line[0:len(line)-1]
```

```
    colunas = line.split(';')
```

```
    IDUtente = eval(colunas[0])
```

```
    Nome = colunas[1].encode()
```

```
    NumeroDispositivoEletronico = colunas[2].encode()
```

```
    #print(IDUtente, Nome, NumeroDispositivoEletronico)
```

```
    donoBinario = donoFormato.pack(IDUtente, Nome, NumeroDispositivoEletronico)
```

```
    f_bin.write(donoBinario)
```

```
print('Fim de conversão')
```

```
f_bin.close()
```

```
f_texto.close()
```



# ConverteDonosTextoEmDonosBinario():Demonstração

```
1 - Inserir dono (.txt)
2 - Alterar dono (.txt)
3 - Eliminar dono (.txt)
4 - Listar todos os donos (.txt)
5 - Listar todos os donos (.bin)
6 - Pesquisa por Nome (.txt)
7 - Pesquisa por Nome e Quantidade (.txt)
8 - Converter donos2.txt em donos2.bin

0 - Voltar

?8
Fim de conversão
```

```
1 - Inserir dono (.txt)
2 - Alterar dono (.txt)
3 - Eliminar dono (.txt)
4 - Listar todos os donos (.txt)
5 - Listar todos os donos (.bin)
6 - Pesquisa por Nome (.txt)
7 - Pesquisa por Nome e Quantidade (.txt)
8 - Converter donos2.txt em donos2.bin

0 - Voltar

?8
Fim de conversão
```





# Gestão de carros: Algoritmo

## Início:

ESCREVER(1 – Inserir(txt)  
ESCREVER(2 – Inserir(bin)  
ESCREVER(3 – Alterar(txt)  
ESCREVER(4 – Eliminar(txt)  
ESCREVER(5 – Listar todos (txt)  
ESCREVER(6 – Listar todos (bin)  
ESCREVER(7 - Pesquisa por Marca(txt)  
ESCREVER(8 - Pesquisa por Marca e quantidade(txt)  
ESCREVER(9 – Converter carros texto em carros binário)  
ESCREVER()  
ESCREVER(0 - Voltar)  
ESCREVER()

Se (0) então  
    voltar atrás  
SeNão (1) então  
    Inserircarro()  
SeNão (2) então  
    inserircarrobin()  
SeNão (3) então  
    ReadAllLinesVectorAlterarCarro()  
SeNão (4) então  
    ReadAllLinesVectorEliminarCarro()  
SeNão (5) então  
    listatodoscarros()  
SeNão (6) então  
    ListarCarrosbin()  
SeNão (7) então  
    pesquisapormarca()  
SeNão (8)então  
    pesquisaporMarcaEQuantidade()  
SeNão (9)então  
    ConverteCarrosTextoEmCarrosBinario()  
SeNão:  
    ESCREVER(Escolha uma opção  
    contida na lista apresentada)  
FimSe  
FimSe  
**Fim**



# Gestão de carros: Python

```
def gestaodecarros ():  
    while True:  
        print("1 - Inserir carro (.txt)")  
        print("2 - Inserir carro (.bin)")  
        print("3 - Alterar carro (.txt)")  
        print("4 - Eliminar carro (.txt)")  
        print("5 - Listar Todos os carros (.txt)")  
        print("6 - Listar Todos os carros (.bin)")  
        print("7 - Pesquisa por Marca ou modelo (.txt)")  
        print("8 - Pesquisa por Marca ou modelo e Quantidade (.txt)")  
        print("9 - Converter carros.txt em carros.bin")  
        print()  
        print("0 - Voltar")  
        print()
```

```
op=input("?")  
if op=="0":  
    break  
elif op=="1":  
    inserircarro()  
elif op=="2":  
    inserircarrobin()  
elif op=="3":  
    ReadAllLinesVectorAlterarCarro()  
elif op=="4":  
    ReadAllLinesVectorEliminarCarro()  
elif op=="5":  
    listatodoscarros()  
elif op=="6":  
    ListarCarrosbin()  
elif op=="7":  
    pesquisapormarca()  
elif op=="8":  
    pesquisaporMarcaEQuantidade()  
elif op=="9":  
    ConverteCarrosTextoEmCarrosBinario()  
else:  
    print("Escolha uma opção contida na lista  
apresentada")
```



# Gestão de carros: Demonstração

```
?2
1 - Inserir carro (.txt)
2 - Inserir carro (.bin)
3 - Alterar carro (.txt)
4 - Eliminar carro (.txt)
5 - Listar Todos os carros (.txt)
6 - Listar Todos os carros (.bin)
7 - Pesquisa por Marca ou modelo (.txt)
8 - Pesquisa por Marca ou modelo e Quantidade (.txt)
9 - Converter carros.txt em carros.bin
0 - Voltar
```

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens
```

```
0 - Terminar
```

```
?2
1 - Inserir carro (.txt)
2 - Inserir carro (.bin)
3 - Alterar carro (.txt)
4 - Eliminar carro (.txt)
5 - Listar Todos os carros (.txt)
6 - Listar Todos os carros (.bin)
7 - Pesquisa por Marca ou modelo (.txt)
8 - Pesquisa por Marca ou modelo e Quantidade (.txt)
9 - Converter carros.txt em carros.bin
```

```
0 - Voltar
```

---



# Inserircarro():Algoritmo

Algoritmo: Inserir\_Carro

Objetivo: Permite inserir um carro no ficheiro de texto carro.txt.

Constantes: FICHEIRO\_CARROSNOVO (TEXTO 40) - Nome do ficheiro dos dados dos carros (Valor: carrosNOVO.txt)

Variáveis

Entrada:

Matricula	(TEXTO 8)	- Matricula do carro	
Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
Modelo	(TEXTO 30)	- Modelo do carro	(Letras de A-Z)
Ano	(TEXTO 4)	- Ano do carro	(>1900, < 3000)
IDUtenteFK	(TEXTO 9)	- Dono do carro	(>=1,<=999999999)
IDCarro	(TEXTO 6)	- Numero do Carro	(>=0,<= 999999)

Saída:

Data: 2016-11-21

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro  
carrosNOVO.txt para escrita  
no fim

ESCREVER os dados no  
ficheiro

FECHAR o ficheiro  
Fim



# Inserircarro():Python

```
def inserircarro():
```

```
    Matricula = input("Matricula ?")
```

```
    Marca = input("Marca ?")
```

```
    Modelo = input("Modelo ?")
```

```
    Ano = input("Ano ?")
```

```
    IDUtenteFK = input("IDUtenteFK ?")
```

```
    IDCarro = input("IDCarro ?")
```

```
    ref_ficheiro = open("carrosNOVO.txt", "at")
```

```
    print(Matricula, ";", Marca, ";", Modelo, ";", Ano, ";", IDUtenteFK, ";", IDCarro, ";", file=ref_ficheiro)
```

```
    ref_ficheiro.close()
```





# Inserircarro():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade
```

```
0 - Voltar
```

```
?1
```

```
Matricula ?43-21-EA
```

```
Marca ?Bmw
```

```
Modelo ?X3
```

```
Ano ?2016
```

```
IDUtenteFK ?12345678
```

```
IDCarro ?321
```

```
1 - Inserir
```

```
2 - Alterar
```

```
3 - Eliminar
```

```
4 - Listar Todos
```

```
5 - Pesquisa por Marca ou modelo
```

```
6 - Pesquisa por Marca ou modelo e Quantidade
```

```
0 - Voltar
```

```
?_
```

```
----- .
```

```
1 - Inserir
```

```
2 - Alterar
```

```
3 - Eliminar
```

```
4 - Listar Todos
```

```
5 - Pesquisa por Marca ou modelo
```

```
6 - Pesquisa por Marca ou modelo e Quantidade
```

```
0 - Voltar
```

```
?1
```

```
Matricula ?12-34-AE
```

```
Marca ?Bmw
```

```
Modelo ?X3
```

```
Ano ?2016
```

```
IDUtenteFK ?123456789
```

```
IDCarro ?123
```



# inserircarrobin():Algoritmo

Algoritmo: Inserir\_Carro

Objetivo: Permite inserir um carro no ficheiro de texto carro.bin.

Constantes: FICHEIRO\_CARROS (binario) - Nome do ficheiro dos dados dos carros (Valor: carros.bin)

Variáveis

Entrada:

Matricula	(TEXTO 8)	- Matricula do carro	
Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
Modelo	(TEXTO 30)	- Modelo do carro	(Letras de A-Z)
Ano	(TEXTO 4)	- Ano do carro	(>1900, < 3000)
IDUtenteFK	(TEXTO 9)	- Dono do carro	(>=1,<=999999999)
IDCarro	(TEXTO 6)	- Numero do Carro	(>=0,<= 999999)

Saída:

Data: 2016-11-21

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro carros.bin  
para escrita no fim

ESCREVER os dados no  
ficheiro

FECHAR o ficheiro  
Fim



# inserircarrobin():Python

```
def inserircarrobin():  
    import struct  
    carroFormato = struct.Struct('15s20s20sii');  
    Matricula = input ('Matricula ?')  
    Marca = input ('Marca ?');  
    Modelo = input ('Modelo ?');  
    Ano = eval(input ('Ano ?'));  
    IDUtenteFK = eval(input ('IDUtenteFK ?'))  
    IDCarro = eval(input ('IDCarro ?'))  
    #           12345678901234567890  
    Matricula = Matricula.encode() ##### '69-20'AO'  
    Marca = Marca.encode()  
    Modelo = Modelo.encode()  
  
    carroBinario= carroFormato.pack(Matricula, Marca, Modelo, Ano, IDUtenteFK, IDCarro)  
  
    f = open("carros.bin2", "ab") ## escrita, fim  
    f.write(carroBinario)  
    f.close()  
    print('Carro inserido com sucesso.')
```



# inserircarrobin():Demonstração

```
C:\WINDOWS\py.exe
1 - Inserir carro (.txt)
2 - Inserir carro (.bin)
3 - Alterar carro (.txt)
4 - Eliminar carro (.txt)
5 - Listar Todos os carros (.txt)
6 - Listar Todos os carros (.bin)
7 - Pesquisa por Marca ou modelo (.txt)
8 - Pesquisa por Marca ou modelo e Quantidade (.txt)
9 - Converter carros.txt em carros.bin
0 - Voltar

?2
Matricula ?01-33-AZ
Marca ?Bmw
Modelo ?120
Ano ?2008
IDUtenteFK ?9262565
IDCarro ?466
Carro inserido com sucesso.
```

- 1 - Inserir carro (.txt)
- 2 - Inserir carro (.bin)
- 3 - Alterar carro (.txt)
- 4 - Eliminar carro (.txt)
- 5 - Listar Todos os carros (.txt)
- 6 - Listar Todos os carros (.bin)
- 7 - Pesquisa por Marca ou modelo (.txt)
- 8 - Pesquisa por Marca ou modelo e Quantidade (.txt)
- 9 - Converter carros.txt em carros.bin

0 - Voltar

?2

Matricula ?12-34-CS

Marca ?Citroen

Modelo ?C4

Ano ?2006

IDUtenteFK ?29656626

IDCarro ?645

Carro inserido com sucesso.



# ReadAllLinesVectorAlterarCarro():Algoritmo

Algoritmo: alterar\_carro

Objetivo: Permite alterar os dados de um carro à escolha.

Constantes: FICHEIRO\_CARROS (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: carros.txt)

Variáveis:

Entrada:

Matricula	(TEXTO 8)	- Matricula do carro	
Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
Modelo	(TEXTO 30)	- Modelo do carro	(Letras de A-Z)
Ano	(TEXTO 4)	- Ano do carro	(>1900, < 3000)
IDUtenteFK	(TEXTO 9)	- Dono do carro	(>=1,<=999999999)
IDCarro	(TEXTO 6)	- Numero do Carro	(>=0,<= 999999)

Saída:

Início:  
Ler dados

ABRIR ficheiro  
carros.txt

ESCREVER os dados  
no ficheiro carros.txt

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# ReadAllLinesVectorAlterarCarro():Python (pt1)

```
def ReadAllLinesVectorAlterarCarro():
```

```
    fname = 'carros.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchMatricula = input("Matricula a alterar?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 6):
```

```
            continue
```

```
        Matricula = colunas[0]
```

```
        Marca = colunas[1]
```

```
        Modelo = colunas[2]
```

```
        Ano = colunas[3]
```

```
        IDUtenteFK = colunas[4]
```

```
        IDCarro = colunas[5]
```

```
    if (SearchMatricula == Matricula):
```

```
        pos = i
```

```
        print ('\n' * 3)
```

```
        print ('Carro encontrado na posição ', i)
```

```
        print ('\n' * 2)
```

```
        print ('Matricula. :', Matricula)
```

```
        print ('Marca.....:', Marca)
```

```
        print ('Modelo....:', Modelo)
```

```
        print ('Ano.....:', Ano)
```

```
        print ('IDUtenteFK:', IDUtenteFK)
```

```
        print ('IDCarro...:', IDCarro)
```

```
        op = input("Alterar ? (s/n) ?")
```



# ReadAllLinesVectorAlterarCarro():Python (pt2)

```
if (op == 's'):
    print('Alterar ')
    Matricula = input("Matricula ?")
    Marca      = input("Marca ?")
    Modelo    = input("Modelo ?")
    Ano        = eval(input("Ano"))
    IDUtenteFK = eval(input("Número do dono ?"))
    IDCarro    = eval(input("Número do carro"))
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
        else:
            print(Matricula, Marca, Modelo, Ano, IDUtenteFK, IDCarro, sep=';', file=outfile)
    outfile.close();
    break;
```

```
if (pos == -1):
    print('Esse carro não existe')
```



# ReadAllLinesVectorAlterarCarro():Demonstração

```
?2
Registos: 9999
Matricula a alterar?47-LJ-22

Carro encontrado na posição 108

Matricula. : 47-LJ-22
Marca..... : BMW
Modelo.... : 530
Ano..... : 1990
IDUtenteFK : 3117403
IDCarro... : 110

Alterar ? (s/n) ?s
Alterar
Matricula ?22-LJ-47
Marca ?Seat
Modelo ?Ibiza
Ano2008
Número do dono ?36541289
Número do carro564
```

```
*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help

?2
Registos: 9999
Matricula a alterar?12-34-AE
Esse carro não existe
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade

0 - Voltar

?2
Registos: 9999
Matricula a alterar?67-UB-49

Carro encontrado na posição 53

Matricula. : 67-UB-49
Marca..... : Citroen
Modelo.... : C5
Ano..... : 1990
IDUtenteFK : 7527845
IDCarro... : 55

Alterar ? (s/n) ?s
Alterar
Matricula ?HJ-54-32
Marca ?Fiat
Modelo ?500
Ano2017
Número do dono ?1255684
Número do carro65

Ln: 20562 Col: 0
```





# ReadAllLinesVectorEliminarCarro():Algoritmo

Algoritmo: eliminar\_carro

Objetivo: Permite eliminar os dados de um carro à escolha.

Constantes: FICHEIRO\_CARROS (TEXTO 90) - Nome do ficheiro dos dados dos carros (Valor: carros.txt)

Variáveis:

Entrada:

Matricula (TEXTO 8) - Matricula do carro (Letras e números)

Saída:

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

Ler dados

ABRIR ficheiro carros.txt

ELIMINAR os dados do  
ficheiro carros.txt referentes  
à matrícula pesquisada

FECHAR o ficheiro

Fim



# ReadAllLinesVectorEliminarCarro():Python(pt1)

```
def ReadAllLinesVectorEliminarCarro():
```

```
    fname = 'carros.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchMatricula = input("Matricula ?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 6):
```

```
            continue
```

```
        Matricula = colunas[0]
```

```
        Marca = colunas[1]
```

```
        Modelo = colunas[2]
```

```
        Ano = colunas[3]
```

```
        IDUtenteFK = colunas[4]
```

```
        IDCarro = colunas[5]
```

```
        if (SearchMatricula == Matricula):
```

```
            pos = i
```

```
            print ('\n' * 3)
```

```
            print ('Carro encontrado na posição ', i)
```

```
            print ('\n' * 2)
```

```
            print ('Matricula. :', Matricula)
```

```
            print ('Marca..... :', Marca)
```

```
            print ('Modelo.... :', Modelo)
```

```
            print ('Ano..... :', Ano)
```

```
            print ('IDUtenteFK :', IDUtenteFK)
```

```
            print ('IDCarro... :', IDCarro)
```

```
            op = input("Eliminar ? (s/n) ?")
```



# ReadAllLinesVectorEliminarCarro():Python(pt2)

```
if (op == 's'):
    print('Eliminando ... ')
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
    outfile.close();
    break;
if (pos == -1):
    print('Esse carro não existe')
```



# ReadAllLinesVectorEliminarCarro():Demonstração

C:\WINDOWS\py.exe

```
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade

0 - Voltar
```

?3

```
Registos: 9998
Matricula ?49-KB-64
```

Carro encontrado na posição 180

```
Matricula. : 49-KB-64
Marca..... : Porsche
Modelo.... : Cheyenne
Ano..... : 2008
IDUtenteFK : 11306225
IDCarro... : 183
```

```
Eliminar ? (s/n) ?s
Eliminando ...
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade
```

0 - Voltar

?3

```
Registos: 9999
Matricula ?HJ-54-32
```

Carro encontrado na posição 53

```
Matricula. : HJ-54-32
Marca..... : Fiat
Modelo.... : 500
Ano..... : 2017
IDUtenteFK : 1255684
IDCarro... : 65
```

```
Eliminar ? (s/n) ?s
Eliminando ...
```



# listatodoscarros():Algoritmo

Algoritmo: listar\_todos\_carros

Objetivo: Permite listar todos os carros.

Constantes: FICHEIRO\_CARROS (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: carros.txt)

Variáveis:

Entrada:

Saída:

Matricula	(TEXTO 8)	- Matricula do carro	
Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
Modelo	(TEXTO 30)	- Modelo do carro	(Letras de A-Z)
Ano	(TEXTO 4)	- Ano do carro	(>1900, < 3000)
IDUtenteFK	(TEXTO 9)	- Dono do carro	(>=1,<=9999999999)
IDCarro	(TEXTO 6)	- Numero do Carro	(>=0,<= 999999)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

ABRIR ficheiro  
carros.txt

ESCREVER os dados  
do ficheiro carros.txt

FECHAR o ficheiro  
Fim.



# listatodoscarros():Python

```
def listatodoscarros():  
    fbrand = 'carros.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        Matricula= colunas[0]  
        Marca= colunas[1]  
        Modelo= colunas[2]  
        Ano= colunas[3]  
        IDUtenteFK= colunas[4]  
        IDCarro= colunas[5]  
        print ("{0:15}{1:^15}{2:^15}{3:^15}{4:^15}{5:>15}".format(Matricula,Marca,Modelo,Ano,IDUtenteFK,IDCarro))  
    infile.close()
```



# listatodoscarros():Demonstração

C:\WINDOWS\py.exe

53-PS-63	Porche	Cheyenne	1990	11245108	1010
72-QD-16	Citroen	C4	2000	4886408	1011
83-UR-33	Peugeot	205	1994	2312597	1012
03-GK-58	BMW	530	2004	3642867	1013
69-NS-66	Renault	Megane	1995	13048821	1014
74-BL-87	Citroen	C3	2001	3700431	1015
31-TV-03	Fiat	600	2010	6410673	1016
83-CL-39	Porche	Carrera	1997	8490644	1017
60-HK-02	Peugeot	305	2002	1622588	1018
49-NP-12	Citroen	C3	2009	4955610	1019
59-MF-59	Bentley	Continental	1999	10392309	1020
79-RQ-87	Fiat	Punto	1998	14544857	1021
08-OE-62	BMW	320	1993	7328099	1022
28-GJ-37	Seat	Ibiza	1998	13272516	1023
87-VS-87	BMW	X3	2006	9848196	1024
58-HO-58	Audi	A2	2009	9256372	1025
31-MS-60	BMW	530	2009	13654432	1026
02-UA-58	Fiat	600	1990	5102232	1027
86-PK-06	Audi	A6	2000	13917071	1028
11-TE-90	Peugeot	305	1990	14448453	1029
95-DD-40	Audi	A6	2003	9773232	1030
14-TC-32	Renault	Kangoo	2009	7537589	1031
29-MN-72	Audi	A3	2002	3895769	1032
03-UC-97	Audi	A6	1996	3590188	1033
69-VC-71	BMW	520	1997	8864299	1034
75-MT-64	Peugeot	305	1996	9841429	1035
71-VA-65	BMW	X3	1993	14682962	1036
25-FP-81	Renault	Kangoo	1992	14108913	1037
00-ON-59	Fiat	Panda	2001	9241189	1038
48-PL-03	Audi	A6	1998	6121746	1039
82-BB-04	Seat	Ibiza	2004	9521722	1040
92-RJ-86	Seat	Ibiza	2010	12718617	1041
45-QA-88	Mitsubishi	Lancer	1999	9937226	1042
45-US-75	Renault	Espace	1994	11442128	1043
67-NK-54	Audi	A3	1993	2869509	1044
38-OJ-08	Audi	A4	2000	6420646	1045
10-KL-24	Citroen	C3	2000	3778431	1046
07-ET-10	Audi	A6	2010	8644515	1047

\*Python 3.6.0 Shell\*

File Edit Shell Debug Options Window Help

409				
75-UC-41	Ferrari	Testarrosa	2010	12078764
410				
34-SU-29	Seat	Ibiza	2007	8489803
411				
91-FP-10	Audi	A3	2007	6950513
412				
30-TQ-09	Audi	A2	2000	13368161
413				
74-MF-34	Citroen	C1	2007	7398414
414				
73-KT-61	Mitsubishi	Lancer	1995	13873979
415				
45-GU-31	Citroen	C2	1990	11878512
416				
79-HN-21	Citroen	C4	1994	8385060
417				
45-MQ-31	Audi	TT	1991	12333874
418				
25-PR-12	BMW	X3	1993	3484052
419				
01-OM-16	Audi	A8	1996	9825897
420				
92-KS-33	Fiat	Panda	1995	4722167
421				
60-CB-74	Bentley	Continental	2010	11449219
422				
85-IF-45	Citroen	C4	2004	5332688
423				
30-JH-37	Porche	Cheyenne	1991	8221403
424				
41-CL-32	Citroen	C5	1990	11559118
425				
72-VM-70	Audi	A2	2002	9483384
426				
64-VS-18	Audi	A2	1995	6594237
427				
58-WN-50	Seat	Ibiza	1990	14184817



# ListarCarrosbin():Algoritmo

Algoritmo: listar\_todos\_carros

Objetivo: Permite listar todos os carros.

Constantes: FICHEIRO\_CARROS (binário) - Nome do ficheiro dos dados dos donos (Valor: carros.bin)

Variáveis:

Entrada:

Saída:

Matricula	(TEXTO 8)	- Matricula do carro	
Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
Modelo	(TEXTO 30)	- Modelo do carro	(Letras de A-Z)
Ano	(TEXTO 4)	- Ano do carro	(>1900, < 3000)
IDUtenteFK	(TEXTO 9)	- Dono do carro	(>=1,<=9999999999)
IDCarro	(TEXTO 6)	- Numero do Carro	(>=0,<= 999999)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

ABRIR ficheiro  
carros.bin

ESCREVER os dados  
do ficheiro carros.bin

FECHAR o ficheiro  
Fim.





# ListarCarrosbin():Python

```
def ListarCarrosbin():  
    import struct  
    carroFormato = struct.Struct('15s20s20sii');  
    f_bin = open("carros.bin", "rb") ##leitura, binario  
    f_bin.seek(0, 2)  
    r = int(f_bin.tell() / carroFormato.size)  
    f_bin.seek(0, 0)  
    pos = -1  
    for i in range(0, r):  
        carroBinario = f_bin.read(carroFormato.size)  
        Matricula, Marca, Modelo, Ano, IDUtenteFK, IDCarro = carroFormato.unpack(carroBinario)  
        Matricula = Matricula.decode()  
        Marca = Marca.decode()  
        Modelo = Modelo.decode()  
        print ("{0:10}".format(Matricula), end="")  
        print ("{0:15}".format(Marca), end="")  
        print ("{0:15}".format(Modelo), end="")  
        print ("{0:10}".format(Ano), end="")  
        print ("{0:12}".format(IDUtenteFK), end="")  
        print ("{0:8}".format(IDCarro), end="")  
    f_bin.close()
```



# ListarCarrosbin():Demonstração

C:\WINDOWS\py.exe

53-PS-63	Porche	Cheyenne	1990	11245108	1010
72-QD-16	Citroen	C4	2000	4886408	1011
83-UR-33	Peugeot	205	1994	2312597	1012
03-GK-58	BMW	530	2004	3642867	1013
69-NS-66	Renault	Megane	1995	13048821	1014
74-BL-87	Citroen	C3	2001	3700431	1015
31-TV-03	Fiat	600	2010	6410673	1016
83-CL-39	Porche	Carrera	1997	8490644	1017
60-HK-02	Peugeot	305	2002	1622588	1018
49-NP-12	Citroen	C3	2009	4955610	1019
59-MF-59	Bentley	Continental	1999	10392309	1020
79-RQ-87	Fiat	Punto	1998	14544857	1021
08-OE-62	BMW	320	1993	7328099	1022
28-GJ-37	Seat	Ibiza	1998	13272516	1023
87-VS-87	BMW	X3	2006	9848196	1024
58-HO-58	Audi	A2	2009	9256372	1025
31-MS-60	BMW	530	2009	13654432	1026
02-UA-58	Fiat	600	1990	5102232	1027
86-PK-06	Audi	A6	2000	13917071	1028
11-TE-90	Peugeot	305	1990	14448453	1029
95-DD-40	Audi	A6	2003	9773232	1030
14-TC-32	Renault	Kangoo	2009	7537589	1031
29-MN-72	Audi	A3	2002	3895769	1032
03-UC-97	Audi	A6	1996	3590188	1033
69-VC-71	BMW	520	1997	8864299	1034
75-MT-64	Peugeot	305	1996	9841429	1035
71-VA-65	BMW	X3	1993	14682962	1036
25-FP-81	Renault	Kangoo	1992	14108913	1037
00-ON-59	Fiat	Panda	2001	9241189	1038
48-PL-03	Audi	A6	1998	6121746	1039
82-BB-04	Seat	Ibiza	2004	9521722	1040
92-RJ-86	Seat	Ibiza	2010	12718617	1041
45-QA-88	Mitsubishi	Lancer	1999	9937226	1042
45-US-75	Renault	Espace	1994	11442128	1043
67-NK-54	Audi	A3	1993	2869509	1044
38-OJ-08	Audi	A4	2000	6420646	1045
10-KL-24	Citroen	C3	2000	3778431	1046
03-GT-18	Audi	A6	2010	8644515	1047

Matricula	Marca	Modelo	Ano	IDUtenteFK
IDCarro				
44-JH-69	Mitsubishi	Colt	1995	9670417
2				
74-RF-54	Mitsubishi	Colt	2006	12312079
3				
80-SU-31	Citroen	C2	1996	14253705
5				
00-NU-80	Audi	TT	2004	4611916
6				
10-PN-70	Audi	A8	1994	11857824
7				
75-FD-04	Porche	Cheyenne	1994	4714863
8				
94-EE-27	Peugeot	305	2005	13581579
9				
32-SV-07	Peugeot	305	1992	8931662
10				
36-KF-05	Opel	Corssa	1994	11190053
11				
99-NK-22	Audi	A2	2007	14101151
12				
70-PO-86	Opel	Kadet	2001	10334539
13				
59-MQ-65	Opel	Kadet	2000	5891832
14				
14-UM-76	Audi	A6	1990	5813959
15				
61-MC-72	Seat	Ibiza	1999	12736971
16				



# pesquisapormarca():Algoritmo

Algoritmo: pesquisa\_por\_marca

Objetivo: Permite pesquisar a marca de um carro

Constantes: FICHEIRO\_CARROS (TEXTO 90) - Nome do ficheiro dos dados dos donos (Valor: carros.txt)

Variáveis:

Entrada:

Marca: (TEXTO 30) - Marca de carro (Letras de A-Z)

Saída:

Matricula (TEXTO 8) - Matricula do carro

Marca (TEXTO 30) - Marca de carro (Letras de A-Z)

Modelo (TEXTO 30) - Modelo do carro (Letras de A-Z)

Ano (TEXTO 4) - Ano do carro (>1900, < 3000)

IDUtenteFK (TEXTO 9) - Dono do carro (>=1, <=999999999)

IDCarro (TEXTO 6) - Numero do Carro (>=0, <= 999999)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro carros.txt

ESCREVER os carros com a  
marca pesquisada

FECHAR o ficheiro  
Fim.



# pesquisapormarca():Python

```
def pesquisapormarca():  
    marcaProcurar = input("Diga a Marca a procurar?")  
    fbrand = 'carros.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        Matricula= colunas[0]  
        Marca= colunas[1]  
        Modelo= colunas[2]  
        Ano= colunas[3]  
        IDUtenteFK= colunas[4]  
        IDCarro= colunas[5]  
        if (Marca.find(marcaProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}{4:^15}{5:>15}".format(Matricula,Marca,Modelo,Ano,IDUtenteFK,IDCarro))  
    infile.close()
```



# pesquisapormarca():Demonstração

```
C:\WINDOWS\py.exe
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade
0 - Voltar

?5
Diga a Marca a procurar?Ferrari
41-AO-04      Ferrari  Testarrosa  2001      3375681      56
70-NJ-81      Ferrari  Testarrosa  1999      7014843      78
47-ML-79      Ferrari  Testarrosa  2005      9121744      92
98-GV-37      Ferrari  Testarrosa  2009      14527721     120
06-CU-97      Ferrari  Testarrosa  1990      7627203      174
01-RG-12      Ferrari  Testarrosa  1995      5418974      275
77-AR-17      Ferrari  Testarrosa  2006      2353781      301
52-MT-66      Ferrari  Testarrosa  1994      12961712     375
75-UC-41      Ferrari  Testarrosa  2010      12078764     410
04-UJ-98      Ferrari  Testarrosa  1992      1995673      445
63-KG-55      Ferrari  Testarrosa  1990      6187349      530
64-UA-08      Ferrari  Testarrosa  1998      6807761      573
95-ID-96      Ferrari  Testarrosa  2008      1187890      659
19-MH-04      Ferrari  Testarrosa  2010      13348804     725
47-RP-03      Ferrari  Testarrosa  1997      4890467      756
80-JM-15      Ferrari  Testarrosa  2006      1394963      906
69-AF-24      Ferrari  Testarrosa  2010      14599744     929
41-MN-79      Ferrari  Testarrosa  2003      11124729     962
65-VF-94      Ferrari  Testarrosa  2001      4416658      965
82-OP-94      Ferrari  Testarrosa  1999      10414537     989
61-SO-28      Ferrari  Testarrosa  1997      4613580      991
68-OO-16      Ferrari  Testarrosa  1997      6308035     1066
40-NC-44      Ferrari  Testarrosa  2009      1789360     1146
47-TS-12      Ferrari  Testarrosa  1994      8103561     1152
00-SO-94      Ferrari  Testarrosa  1995      8305925     1165
82-VT-64      Ferrari  Testarrosa  1995      8333550     1249
00-DK-66      Ferrari  Testarrosa  1994      7070707     1273
```

```
*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade
0 - Voltar

?5
Diga a Marca a procurar?Audi
00-NU-80      Audi      TT      2004      4611916
6
10-PN-70      Audi      A8      1994      11857824
7
99-NK-22      Audi      A2      2007      14101151
12
14-UM-76      Audi      A6      1990      5813959
15
76-PR-80      Audi      A1      2002      7225106
18
10-LF-55      Audi      A2      1995      9344066
21
37-CS-44      Audi      A8      2006      13264408
22
00-JB-43      Audi      A5      1996      10622351
23
84-AC-17      Audi      A3      1990      7755917
27
67-DN-08      Audi      A1      2004      7677588
40
94-LI-85      Audi      A8      2006      6514918
49
11-AO-50      Audi      A4      2006      12137600
59
75-AB-92      Audi      A2      1997      9969680
62
65-JC-56      Audi      A4      2006      13600935
64
38-KF-43      Audi      A3      1996      6441671

Ln: 32687 Col: 1
```



# pesquisaporMarcaEQuantidade():Algoritmo

Algoritmo: pesquisa\_por\_marca\_e\_quantidade

Objetivo: Permite saber quantos carros há da mesma marca.

Constantes: FICHEIRO\_CARROS (TEXTO 90) - Nome do ficheiro dos dados dos carros(Valor: carros.txt)

Variáveis:

Entrada:

Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
-------	------------	------------------	-----------------

Saída:

Marca	(TEXTO 30)	- Marca de carro	(Letras de A-Z)
-------	------------	------------------	-----------------

n	(TEXTO 3)	- Numero carros da mesma marca	( $\geq 0, \leq 999$ )
---	-----------	--------------------------------	------------------------

Início:

LER dados

ABRIR ficheiro carros.txt

ESCREVER os carros com  
a marca pesquisada

ESCREVER o numero de  
carros com a marca  
pesquisada

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# pesquisaporMarcaEQuantidade():Python

```
def pesquisaporMarcaEQuantidade():  
    marcaProcurar = input("Diga a Marca a procurar?")  
    fbrand = 'carros.txt'  
    infile = open(fbrand, "r")  
    n = 0  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        Matricula= colunas[0]  
        Marca= colunas[1]  
        Modelo= colunas[2]  
        Ano= colunas[3]  
        IDUtenteFK= colunas[4]  
        IDCarro= colunas[5]  
        if (Marca==marcaProcurar):  
            n = n + 1  
    print("Numero de pessoas com a marca",marcaProcurar, n)  
    infile.close()
```



# pesquisaporMarcaEQuantidade():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
5 - Pesquisa por Marca ou modelo e Quantidade
0 - Voltar

?6
Diga a Marca a procurar?Citroen
Numero de pessoas com a marca Citroen 1318
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por Marca ou modelo
6 - Pesquisa por Marca ou modelo e Quantidade
0 - Voltar

?6
Diga a Marca a procurar?Opel
Numero de pessoas com a marca Opel 538
```



# ConverteCarrosTextoEmCarrosBinario():Python

```
def ConverteCarrosTextoEmCarrosBinario():
```

```
    import struct
```

```
    carroFormato = struct.Struct('15s20s20siii');
```

```
    f_texto = open('carros.txt', "r")
```

```
    f_bin = open("carros.bin", "wb")
```

```
    lines = f_texto.readlines();
```

```
    for i in range(2, len(lines)):
```

```
        line = lines[i]
```

```
        line = line[0:len(line)-1]
```

```
        colunas = line.split(';')
```

```
        Matricula = colunas[0].encode()
```

```
        Marca = colunas[1].encode()
```

```
        Modelo = colunas[2].encode()
```

```
        Ano= eval(colunas[3])
```

```
        IDUtenteFK = eval(colunas[3])
```

```
        IDCarro = eval(colunas[4])
```

```
        #print(Matricula, Marca, Modelo, Ano, IDUtenteFK, IDCarro)
```

```
        carroBinario = carroFormato.pack(Matricula, Marca, Modelo, Ano, IDUtenteFK,  
IDCarro)
```

```
        f_bin.write(carroBinario)
```

```
        print('Fim de conversão')
```

```
        f_bin.close()
```

```
        f_texto.close()
```

# ConverteCarrosTextoEmCarrosBinario():Demonstração

```
1 - Inserir carro (.txt)
2 - Inserir carro (.bin)
3 - Alterar carro (.txt)
4 - Eliminar carro (.txt)
5 - Listar Todos os carros (.txt)
6 - Listar Todos os carros (.bin)
7 - Pesquisa por Marca ou modelo (.txt)
8 - Pesquisa por Marca ou modelo e Quantidade (.txt)
9 - Converter carros.txt em carros.bin

0 - Voltar

?9
Fim de conversão
```

```
1 - Inserir carro (.txt)
2 - Inserir carro (.bin)
3 - Alterar carro (.txt)
4 - Eliminar carro (.txt)
5 - Listar Todos os carros (.txt)
6 - Listar Todos os carros (.bin)
7 - Pesquisa por Marca ou modelo (.txt)
8 - Pesquisa por Marca ou modelo e Quantidade (.txt)
9 - Converter carros.txt em carros.bin

0 - Voltar

?9
Fim de conversão
```





# Gestão de cidades: Algoritmo

## Início:

ESCREVER(1 - Inserir)

ESCREVER(2 - Alterar)

ESCREVER(3 - Eliminar)

ESCREVER(4 - Listar Todas)

ESCREVER(5 - Pesquisa por IDCidade)

ESCREVER(6 - Pesquisa por Nome da cidade)

ESCREVER()

ESCREVER(0 - Voltar)

ESCREVER()

Se (0) então

Voltar atrás

SeNão (1)então

inserircidade()

SeNão (2) então

ReadAllLinesVectorAlterarCidade()

SeNão (3) então

ReadAllLinesVectorEliminarCidade()

SeNão (4) então

listatodascidades()

SeNão (5) então

pesquisaporidcidade()

SeNão (6) então

pesquisaporNomeDaCidade()

SeNão:

ESCREVER(Escolha uma opção contida na  
lista apresentada)

FimSe.

FimSe.

**Fim.**



# Gestão de cidades: Python

```
def gestaoDeDistancias ():
```

```
    while True:
```

```
        print("1 - Inserir")
```

```
        print("2 - Alterar")
```

```
        print("3 - Eliminar")
```

```
        print("4 - Listar Todos")
```

```
        print("5 - Pesquisa por cidade de partida")
```

```
        print("6 - Pesquisa por cidade de chegada")
```

```
        print("7 - Pesquisa por distancia")
```

```
        print("8 - Pesquisa por Preço")
```

```
        print()
```

```
        print("0 - Voltar")
```

```
        print()
```

```
op=input("?")
```

```
    if op=="0":
```

```
        break
```

```
    elif op=="1":
```

```
        inserirCidade()
```

```
    elif op=="2":
```

```
        ReadAllLinesVectorAlterarCidade()
```

```
    elif op=="3":
```

```
        ReadAllLinesVectorEliminarCidade()
```

```
    elif op=="4":
```

```
        listatodascidades()
```

```
    elif op=="5":
```

```
        pesquisaporidcidade()
```

```
    elif op=="6":
```

```
        pesquisaporNomeDaCidade()
```

```
    else:
```

```
        print("Escolha uma opção contida na lista apresentada")
```



# Gestão de cidades: Demonstração

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?3
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade
```

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?3
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

d
```



# inserircidade():Algoritmo

Algoritmo: Inserir\_cidade

Objetivo: Permite inserir uma cidade no ficheiro de texto cidadesNOVO.txt.

Constantes: FICHEIRO\_CIDADESNOVO (TEXTO 40) - Nome do ficheiro dos dados das cidades (Valor: cidadesNOVO.txt)

Variáveis

Entrada:

IDCidade (TEXTO 1)	- Numero da cidade	(Números de 1-9)
Cidade (TEXTO 30)	- Nome da cidade	(Letras de A-Z)

Saída:

Data: 2016-11-21

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro  
cidadesNOVO.txt para  
escrita no fim

ESCREVER os dados no  
ficheiro

FECHAR o ficheiro  
Fim.



# inserircidade():Python

```
def inserircidade():
```

```
    IDCidade = eval(input("IDCidade ?"))
```

```
    Cidade = input("Cidade ?")
```

```
    ref_ficheiro = open("cidadesNOVO.txt", "at")
```

```
    print(IDCidade, ";", Cidade, ";",file=ref_ficheiro)
```

```
    ref_ficheiro.close()
```



# inserircidade():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?1
IDCidade ?1
Cidade ?Castelo Branco
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?1
IDCidade ?8
Cidade ?Guarda
```





# ReadAllLinesVectorAlterarCidade():Algoritmo

Algoritmo: alterar\_cidade

Objetivo: Permite alterar os dados de uma cidade à escolha.

Constantes: FICHEIRO\_Cidades (TEXTO 90) - Nome do ficheiro dos dados das cidades (Valor: cidades.txt)

Variáveis:

Entrada:

<u>Saída</u> :	IDCidade (TEXTO 1)	- Numero da cidade	(Números de 1-9)
	Cidade (TEXTO 30)	- Nome da cidade	(Letras de A-Z)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

Ler dados

ABRIR ficheiro  
cidades.txt

ESCREVER os dados  
no ficheiro  
cidades.txt

FECHAR o ficheiro  
Fim.



# ReadAllLinesVectorAlterarCidade():Python(pt1)

```
def ReadAllLinesVectorAlterarCidade():
```

```
    fname = 'cidades.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchIDCidade = input("IDCidade a alterar?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 2):
```

```
            continue
```

```
        IDCidade          = colunas[0]
```

```
        Cidade           = colunas[1]
```

```
    if (SearchIDCidade == IDCidade):
```

```
        pos = i
```

```
        print ('\n' * 3)
```

```
        print ('IDCidade encontrado na posição ', i)
```

```
        print ('\n' * 2)
```

```
        print ('IDCidade.....:', IDCidade)
```

```
        print ('Cidade.....:', Cidade)
```

```
    op = input("Alterar ? (s/n) ?")
```

```
    if (op == 's'):
```



# ReadAllLinesVectorAlterarCidade():Python (pt2)

```
print('Alterar ')
IDCidade = input("IDCidade ?")
Cidade      = input("Cidade ?")
outfile = open(fname, "w")
for i in range(len(lines)):
    if (i != pos):
        line = lines[i]
        line[0:len(line)-1]
        print(line, file=outfile, end="")
    else:
        print(IDCidade, Cidade, sep=';', file=outfile)
outfile.close();
break;
if (pos == -1):
    print('Essa cidade nãoexiste')
```



# ReadAllLinesVectorAlterarCidade():Demonstração

C:\WINDOWS\py.exe

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar
```

?2

Registos: 12

IDCidade a alterar?3

IDCidade encontrado na posição 3

IDCidade..... : 3

Cidade..... : Braga

Alterar ? (s/n) ?s

Alterar

IDCidade ?15

Cidade ?Vila Real

1 - Inserir

2 - Alterar

3 - Eliminar

4 - Listar Todas

5 - Pesquisa por IDCidade

6 - Pesquisa por Nome da cidade

0 - Voltar

?2

Registos: 12

IDCidade a alterar?15

IDCidade encontrado na posição 3

IDCidade..... : 15

Cidade..... : Vila Real

Alterar ? (s/n) ?s

Alterar

IDCidade ?3

Cidade ?Braga



# ReadAllLinesVectorEliminarCidade(): Algoritmo

Algoritmo: eliminar\_cidades

Objetivo: Permite eliminar os dados de uma cidade à escola.

Constantes: FICHEIRO\_CIDADES (TEXTO 90) - Nome do ficheiro dos dados das cidades(Valor: cidades.txt)

Variáveis:

Entrada:

IDCidade (TEXTO 2) - Numero da cidade (Números de 1-9)

Saída:

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

Ler dados

ABRIR ficheiro cidades.txt

ELIMINAR os dados do  
ficheiro cidades.txt referentes  
à cidade pesquisada

FECHAR o ficheiro

Fim.



# ReadAllLinesVectorEliminarCidade():Python

```
def ReadAllLinesVectorEliminarCidade():
```

```
    fname = 'cidades.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchIDCidade = input("IDCidade ?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 2):
```

```
            continue
```

```
        IDCidade          = colunas[0]
```

```
        Cidade            = colunas[1]
```

```
        if (SearchIDCidade == IDCidade):
```

```
            pos = i
```

```
            print ('\n' * 3)
```

```
            print ('IDCidade encontrado na posição ', i)
```

```
            print ('\n' * 2)
```

```
            print ('IDCidade.....:', IDCidade)
```

```
            print ('Cidade.....:', Cidade)
```

```
            op = input("Eliminar ? (s/n) ?")
```

```
            if (op == 's'):
```

```
                print('Eliminando ... ')
```

```
                outfile = open(fname, "w")
```

```
                for i in range(len(lines)):
```

```
                    if (i != pos):
```

```
                        line = lines[i]
```

```
                        line[0:len(line)-1]
```

```
                        print(line, file=outfile, end="")
```

```
                outfile.close();
```

```
                break;
```

```
            if (pos == -1):
```

```
                print('Essa cidade não existe')
```



# ReadAllLinesVectorEliminarCidade():Demonstração

```
1 - Inserir  
2 - Alterar  
3 - Eliminar  
4 - Listar Todas  
5 - Pesquisa por IDCidade  
6 - Pesquisa por Nome da cidade
```

```
0 - Voltar
```

```
?3  
Registos: 12  
IDCidade ?10
```

```
IDCidade encontrado na posição 1
```

```
IDCidade..... : 10  
Cidade..... : Faro
```

```
Eliminar ? (s/n) ?n
```

```
1 - Inserir  
2 - Alterar  
3 - Eliminar  
4 - Listar Todas  
5 - Pesquisa por IDCidade  
6 - Pesquisa por Nome da cidade
```

```
0 - Voltar
```

```
?3  
Registos: 12  
IDCidade ?2
```

```
IDCidade encontrado na posição 2
```

```
IDCidade..... : 2  
Cidade..... : Maia
```

```
Eliminar ? (s/n) ?n
```



# listatodascidades(): Algoritmo

Algoritmo: listar\_todas\_cidades

Objetivo: Permite listar todas as cidades do ficheiro.

Constantes: FICHEIRO\_CIDADES (TEXTO 90) - Nome do ficheiro dos dados das cidades (Valor: cidades.txt)

Variáveis:

Entrada:

Saída:

IDCidade	(TEXTO 1)	- Numero da cidade	(Números de 1-9)
Cidade	(TEXTO 30)	- Nome da cidade	(Letras de A-Z)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

ABRIR ficheiro  
cidades.txt

ESCREVER os dados do  
ficheiro cidades.txt

FECHAR o ficheiro  
Fim.





# listatodascidades():Python

```
def listatodascidades():  
    fbrand = 'cidades.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidade= colunas[0]  
        cidade= colunas[1]  
        print ("{0:15}{1:^15}".format(idcidade,cidade))  
    infile.close()
```



# listatodascidades():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?4
IDCidade      Cidade
10            Faro
2             Maia
3             Braga
4             Aveiro
5             Leiria
6             Torres Novas
7             Santarem
8             Lisboa
9             Setubal
34            Manchester
21            dede

1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?4
IDCidade      Cidade
10            Faro
2             Maia
3             Braga
4             Aveiro
5             Leiria
6             Torres Novas
7             Santarem
8             Lisboa
9             Setubal
34            Manchester
21            dede

1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?
```



# pesquisaporidcidade(): Algoritmo

Algoritmo: pesquisa\_por\_id\_cidade

Objetivo: Permite pesquisar uma cidade através do seu id

Constantes: FICHEIRO\_CIDADES (TEXTO 90) - Nome do ficheiro dos dados das cidades (Valor: cidades.txt)

Variáveis:

Entrada:

IDCidade	(TEXTO 1)	- Numero da cidade	(Números de 1-9)
----------	-----------	--------------------	------------------

Saída:

IDCidade	(TEXTO 1)	- Numero da cidade	(Números de 1-9)
----------	-----------	--------------------	------------------

Cidade	(TEXTO 30)	- Nome da cidade	(Letras de A-Z)
--------	------------	------------------	-----------------

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER id da cidade dado  
ABRIR ficheiro cidades.txt

ESCREVER a cidade  
pesquisada

FECHAR o ficheiro  
Fim.



# pesquisaporidcidade():Python

```
def pesquisaporidcidade():  
    idcidadeProcurar = input("Diga o IDCidade a procurar?")  
    fbrand = 'cidades.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidade= colunas[0]  
        cidade= colunas[1]  
        if (idcidade.find(idcidadeProcurar)>=0):  
            print ("{0:15}{1:^15}".format(idcidade,cidade))  
    infile.close()
```



# pesquisaporidcidade():Demonstração

```
21          gcedc
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?5
Diga o IDCidade a procurar?3
3          Braga
34         Manchester
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade

0 - Voltar

?5
Diga o IDCidade a procurar?10
10         Faro
?          T-----
```



# pesquisaporNomeDaCidade():Algoritmo

Algoritmo: pesquisa\_por\_nome\_cidade

Objetivo: Permite pesquisar uma cidade através do seu nome

Constantes: FICHEIRO\_CIDADES (TEXTO 90) - Nome do ficheiro dos dados das cidades (Valor: cidades.txt)

Variáveis:

Entrada:

Cidade	(TEXTO 30)	- Nome da cidade	(Letras de A-Z)
--------	------------	------------------	-----------------

Saída:

IDCidade	(TEXTO 1)	- Numero da cidade	(Números de 1-9)
----------	-----------	--------------------	------------------

Cidade	(TEXTO 30)	- Nome da cidade	(Letras de A-Z)
--------	------------	------------------	-----------------

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER nome da cidade  
dado

ABRIR ficheiro  
cidades.txt

ESCREVER a cidade  
pesquisada

FECHAR o ficheiro  
Fim.



# pesquisaporNomeDaCidade():Python

```
def pesquisaporNomeDaCidade():  
    cidadeProcurar = input("Diga a Cidade a procurar?")  
    fbrand = 'cidades.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDCidade= colunas[0]  
        cidade= colunas[1]  
        if (cidade.find(cidadeProcurar)>=0):  
            print ("{0:15}{1:^15}".format(IDCidade,cidade))  
    infile.close()
```



# pesquisaporNomeDaCidade():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade
0 - Voltar

?6
Diga a Cidade a procurar?Maia
2          Maia
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por IDCidade
6 - Pesquisa por Nome da cidade
0 - Voltar

?6
Diga a Cidade a procurar?Aveiro
4          Aveiro
```







# Gestão de distâncias: Algoritmo

## Início:

ESCREVER(1 - Inserir)  
ESCREVER(2 - Alterar)  
ESCREVER(3 - Eliminar)  
ESCREVER(4 - Listar Todos)  
ESCREVER(5 - Pesquisa por cidade de partida)  
ESCREVER(6 - Pesquisa por cidade de chegada)  
ESCREVER(7 - Pesquisa por distancia)  
ESCREVER(8 - Pesquisa por Preço)  
ESCREVER()  
ESCREVER(0 - Voltar)  
ESCREVER()

Início:

Se (0) então

Voltar atrás

SeNão (1) então

inserirdistancia()

SeNão (2) então

ReadAllLinesVectorAlterarDistancia()

SeNão (3) então

ReadAllLinesVectorEliminarDistancia()

SeNão (4) então

Listatodasdistancias()

SeNão(5) então

pesquisaporIDcidadeA()

SeNão (6) então

pesquisaporIDcidadeB()

SeNão (7) então

pesquisapordistancia()

SeNão (8) então

pesquisaporpreço()

SeNão:

ESCREVER(Escolha uma opção contida na lista apresentada)

FimSe

FimSe

**Fim**



# Gestão de distâncias: Python

```
def gestaodedistancias ():
```

```
    while True:
```

```
        print("1 - Inserir")
```

```
        print("2 - Alterar")
```

```
        print("3 - Eliminar")
```

```
        print("4 - Listar Todos")
```

```
        print("5 - Pesquisa por cidade de partida")
```

```
        print("6 - Pesquisa por cidade de chegada")
```

```
        print("7 - Pesquisa por distancia")
```

```
        print("8 - Pesquisa por preço")
```

```
        print()
```

```
        print("0 - Voltar")
```

```
        print()
```

```
    op=input("?")
```

```
    if op=="0":
```

```
        break
```

```
    elif op=="1":
```

```
        inserirdistancia()
```

```
    elif op=="2":
```

```
        ReadAllLinesVectorAlterarDistancia()
```

```
    elif op=="3":
```

```
        ReadAllLinesVectorEliminarDistancia()
```

```
    elif op=="4":
```

```
        listatodasdistancias()
```

```
    elif op=="5":
```

```
        pesquisaporIDcidadeA()
```

```
    elif op=="6":
```

```
        pesquisaporIDcidadeB()
```

```
    elif op=="7":
```

```
        pesquisapordistancia()
```

```
    elif op=="8":
```

```
        pesquisaporpreço()
```

```
    else:
```

```
        print("Escolha uma opção contida na lista apresentada")
```



# Gestão de distâncias: Demonstração

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?4
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar

? _
```

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?4
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar

?
```



# inserirdistancia():Algoritmo

Algoritmo: Inserir\_distancia

Objetivo: Permite inserir uma distancia no ficheiro de texto distanciasNOVO.txt.

Constantes: FICHEIRO\_DISTANCIASNOVO (TEXTO 40) - Nome do ficheiro dos dados das distancias (Valor: distanciasNOVO.txt)

Variáveis

Entrada:

IDCidadeA	(TEXTO 1)- Número da cidade	(Números de 1-9)
IDCidadeB	(TEXTO 1)- Número da cidade	(Números de 1-9)
Distancia	(TEXTO 30)- Distância entre as cidades	(>=0,<999)
Preco	(TEXTO 30)- Preço	(>=0,<100)

Saída:

Data: 2016-11-21

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro  
distanciasNOVO.txt  
para escrita no fim

ESCREVER os dados  
no ficheiro

FECHAR o ficheiro  
Fim.



# inserirdistancia():Python

```
def inserirdistancia():  
    IDCidadeA = eval(input("IDCidade A ?"))  
    IDCidadeB = input("IDCidade B ?")  
    Distancia = input("distancia ?")  
    Preco = input("Preço ?")  
    ref_ficheiro = open("distanciasNOVO.txt", "at")  
    print(IDCidadeA, ";", IDCidadeB, ";", Distancia, ";", Preco, ";", file=ref_ficheiro)  
    ref_ficheiro.close()
```

# inserirdistancia():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?1
```

```
IDCidade A ?5
```

```
IDCidade B ?6
```

```
distancia ?100
```

```
Preço ?2
```

```
._._.
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?1
```

```
IDCidade A ?3
```

```
IDCidade B ?10
```

```
distancia ?200
```

```
Preço ?5
```



# ReadAllLinesVectorAlterarDistancia():Algoritmo

Algoritmo: alterar\_distancia

Objetivo: Permite alterar os dados de uma distância à escolha.

Constantes: FICHEIRO\_DISTANCIAS (TEXTO 90) - Nome do ficheiro dos dados das distâncias (Valor: distancias.txt)

Variáveis:

Entrada:

IDCidadeA (TEXTO 1)- Número da cidade (Números de 1-9)

IDCidadeB (TEXTO 1)- Número da cidade (Números de 1-9)

Distancia (TEXTO 30) - Distância entre as cidades ( $\geq 0, < 999$ )

Preço (TEXTO 30) - Preço ( $\geq 0, < 100$ )

Saída:

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

Ler dados

ABRIR ficheiro  
distancias.txt

ESCREVER os dados no  
ficheiro distancias.txt

FECHAR o ficheiro  
Fim.



# ReadAllLinesVectorAlterarDistancia():Python(pt1)

```
def ReadAllLinesVectorAlterarDistancia():
```

```
    fname = 'distancias.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchIDCidadeA = input("IDCidadeA a alterar?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 4):
```

```
            continue
```

```
        IDCidadeA = colunas[0]
```

```
        IDCidadeB = colunas[1]
```

```
        Distancia = colunas[2]
```

```
        Preco = colunas[3]
```

```
        if (SearchIDCidadeA == IDCidadeA):
```

```
            pos = i
```

```
            print ('\n' * 3)
```

```
            print ('IDCidadeA encontrado na posição ', i)
```

```
            print ('\n' * 2)
```

```
            print ('IDCidadeA. :', IDCidadeA)
```

```
            print ('IDCidadeB. :', IDCidadeB)
```

```
            print ('Distancia. :', Distancia)
```

```
            print ('Preço..... :', Preco)
```

```
            op = input("Alterar ? (s/n) ?")
```





# ReadAllLinesVectorAlterarDistancia():Python(pt2)

```
if (op == 's'):
    print('Alterar ')
    IDCidadeA = input("IDCidadeA ?")
    IDCidadeB = input("IDCidadeB ?")
    Modelo   = input("Distancia ?")
    Preco     = eval(input("Preço"))
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
        else:
            print(IDCidadeA, IDCidadeB, Distancia, Preco, sep=';', file=outfile)
    outfile.close();
    break;
```

```
if (pos == -1):
    print('Essa distancia não existe')
```



# ReadAllLinesVectorAlterarDistancia(): Demonstração

```
C:\WINDOWS\py.exe
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar

?2
Registos: 45
IDCidadeA a alterar?3

IDCidadeA encontrado na posição 17

IDCidadeA. : 3
IDCidadeB. : 4
Distancia. : 76.8
Preço..... : 9.22

Alterar ? (s/n) ?s
Alterar
IDCidadeA ?2
IDCidadeB ?4
Distancia ?100
Preço?
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar

?2
Registos: 45
IDCidadeA a alterar?8

IDCidadeA encontrado na posição 42

IDCidadeA. : 8
IDCidadeB. : 9
Distancia. : 45.2
Preço..... : 5.42

Alterar ? (s/n) ?n

IDCidadeA encontrado na posição 43

IDCidadeA. : 8
IDCidadeB. : 10
Distancia. : 297.2
Preço..... : 35.66

Alterar ? (s/n) ?s
Alterar
IDCidadeA ?7
IDCidadeB ?6
Distancia ?545
Preço?12
```



# ReadAllLinesVectorEliminarDistancia():Algoritmo

Algoritmo: eliminar\_distancias

Objetivo: Permite eliminar os dados de uma distância à escolha.

Constantes: FICHEIRO\_DISTANCIAS (TEXTO 90) - Nome do ficheiro dos dados das distancias(Valor: distancias.txt)

Variáveis:

Entrada:

IDCidadeA (TEXTO 1)- Numero da cidade (Números de 1-9)

Saída:

Início:  
Ler dados

ABRIR ficheiro  
distancias.txt

ELIMINAR os dados do  
ficheiro distancias.txt  
referentes à distancia  
pesquisada

FECHAR o ficheiro

Data: 2017-01-02

Versão: 1.0

Obs:



# ReadAllLinesVectorEliminarDistancia():Python(pt1)

```
def ReadAllLinesVectorEliminarDistancia():
```

```
    fname = 'distancias.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchIDCidadeA = input("IDCidadeA ?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 4):
```

```
            continue
```

```
        IDCidadeA = colunas[0]
```

```
        IDCidadeB = colunas[1]
```

```
        Distancia = colunas[2]
```

```
        Preco     = colunas[3]
```

```
    if (SearchIDCidadeA == IDCidadeA):
```

```
        pos = i
```

```
        print ('\n' * 3)
```

```
        print ('IDCidadeA encontrado na posição ', i)
```

```
        print ('\n' * 2)
```

```
        print ('IDCidadeA. :', IDCidadeA)
```

```
        print ('IDCidadeB. :', IDCidadeB)
```

```
        print ('Distancia. :', Distancia)
```

```
        print ('Preço..... :', Preco)
```

```
        op = input("Eliminar ? (s/n) ?")
```



# ReadAllLinesVectorEliminarDistancia():Python(pt2)

```
if (op == 's'):
    print('Eliminando ... ')
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
    outfile.close();
    break;
if (pos == -1):
    print('Essa distancia não existe')
```



# ReadAllLinesVectorEliminarDistancia(): Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?3
Registos: 45
IDCidadeA ?2
```

```
IDCidadeA encontrado na posição 10
```

```
IDCidadeA. : 2
IDCidadeB. : 3
Distancia. : 27.8
Preço..... : 3.34
```

```
Eliminar ? (s/n) ?s
Eliminando ...
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?3
Registos: 44
IDCidadeA ?8
```

```
IDCidadeA encontrado na posição 41
```

```
IDCidadeA. : 8
IDCidadeB. : 9
Distancia. : 45.2
Preço..... : 5.42
```

```
Eliminar ? (s/n) ?s
Eliminando ...
```



# listatodasdistanCIAS():Algoritmo

Algoritmo: listar\_todas\_distancias

Objetivo: Permite listar todas distancias.

Constantes: FICHEIRO\_DISTANCIAS (TEXTO 90) - Nome do ficheiro dos dados das distancias  
(Valor: distancias.txt)

Variáveis:

Entrada:

Saída:

IDCidadeA	(TEXTO 1)- Número da cidade	(Números de 1-9)
IDCidadeB	(TEXTO 1)- Número da cidade	(Números de 1-9)
Distancia	(TEXTO 30)- Distância entre as cidades	(>=0,<999)
Preco	(TEXTO 30)- Preço	(>=0,<100)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

ABRIR ficheiro  
distancias.txt

ESCREVER os dados do  
ficheiro distancias.txt

FECHAR o ficheiro  
Fim.



# listatodasdistancias():Python

```
def listatodasdistancias():  
    fbrand = 'distancias.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidadeA= colunas[0]  
        idcidadeB= colunas[1]  
        distancia= colunas[2]  
        preco= colunas[3]  
        print ("{0:15}{1:^15}{2:^15}{3:^15}".format(idcidadeA,idcidadeB,distancia,preco))  
    infile.close()
```





# listatodasdistanCIAS():Demonstração

C:\WINDOWS\py.exe

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar
```

```
?4
IDCidadeA      IDCidadeB      Distancia      Preco
1              2          12.5          1.5
1              3          35.8          4.3
1              4          42.6          5.11
1              5          156.7         18.8
1              6          189.3         22.72
1              7          214.1         25.69
1              8          296.7         35.6
1              9          326.8         39.22
1             10          650.1         78.01
2              5          167.7         20.12
2              6          197.6         23.71
2              7          215.4         25.85
2              8          278.8         33.46
2              9          313.7         37.64
2             10          652.6         78.31
2              4          76.8          2
2              5          105.8         12.7
2              6          214.2         25.7
2              7          298.3         35.8
2              8          358          42.96
2              9          370          44.4
2             10          696.4         83.57
3              5          48.2          5.78
3              6          100.3         12.04
3              7          165.3         19.84
3              8          198.2         23.78
3              9          243.5         29.22
3             10          502.3         60.28
4              6          59           7.08
4              7          105.2         12.62
4              8          128.9         15.47
4              9          198.5         23.82
4             10          470.2         56.42
5              7          28.9          3.47
5              8          100.2         12.02
5              9          142.5         17.1
5             10          178.9         21.47
6              8          40.1          4.81
6              9          87.2          10.46
6             10          303.2         36.38
7              6          297.2         12
7             10          256.3         30.76
9
```

\*Python 3.6.0 Shell\*

File Edit Shell Debug Options Window Help

Eliminando ...

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?4
IDCidadeA      IDCidadeB      Distancia      Preco
1              2          12.5          1.5
1              3          35.8          4.3
1              4          42.6          5.11
1              5          156.7         18.8
1              6          189.3         22.72
1              7          214.1         25.69
1              8          296.7         35.6
1              9          326.8         39.22
1             10          650.1         78.01
2              5          167.7         20.12
2              6          197.6         23.71
2              7          215.4         25.85
2              8          278.8         33.46
2              9          313.7         37.64
2             10          652.6         78.31
2              4          76.8          2
2              5          105.8         12.7
2              6          214.2         25.7
2              7          298.3         35.8
2              8          358          42.96
2              9          370          44.4
2             10          696.4         83.57
3              5          48.2          5.78
3              6          100.3         12.04
3              7          165.3         19.84
3              8          198.2         23.78
3              9          243.5         29.22
3             10          502.3         60.28
4              6          59           7.08
4              7          105.2         12.62
4              8          128.9         15.47
4              9          198.5         23.82
4             10          470.2         56.42
5              7          28.9          3.47
5              8          100.2         12.02
5              9          142.5         17.1
5             10          178.9         21.47
6              8          40.1          4.81
6              9          87.2          10.46
6             10          303.2         36.38
7              6          297.2         12
7             10          256.3         30.76
- - -
```



# pesquisaporIDcidadeA():Algoritmo

Algoritmo: pesquisa\_por\_id\_cidade\_a

Objetivo: Permite pesquisar por uma cidade de partida.

Constantes: FICHEIRO\_CIDADES (TEXT0 90) - Nome do ficheiro dos dados das cidades (Valor: cidades.txt)

Variáveis:

Entrada:

IDCidadeA (TEXT0 1)- Número da cidade (Números de 1-9)

Saída:

IDCidadeA (TEXT0 1)- Número da cidade (Números de 1-9)

IDCidadeB (TEXT0 1)- Número da cidade (Números de 1-9)

Distância (TEXT0 30)- Distância entre as cidades ( $\geq 0, < 999$ )

Preco (TEXT0 30)- Preço ( $\geq 0, < 100$ )

Início:  
LER dados

ABRIR ficheiro  
cidades.txt

ESCREVER a cidade  
pesquisada

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# pesquisaporIDcidadeA():Python

```
def pesquisaporIDcidadeA():  
    idcidadeAProcurar = input("Diga a cidade de partida a procurar [IDcidadeA]?")  
    fbrand = 'distancias.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidadeA= colunas[0]  
        idcidadeB= colunas[1]  
        distancia= colunas[2]  
        preco= colunas[3]  
        if (idcidadeA.find(idcidadeAProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(idcidadeA,idcidadeB,distancia,preco))  
    infile.close()
```



# pesquisaporIDcidadeA():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
0 - Voltar
```

```
?5
Diga a cidade de partida a procurar [IDcidadeA]?3
3          5          105.8          12.7
3          6          214.2          25.7
3          7          298.3          35.8
3          8           358          42.96
3          9          370          44.4
3         10         696.4          83.57
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?5
Diga a cidade de partida a procurar [IDcidadeA]?1
1          2          12.5          1.5
1          3          35.8          4.3
1          4          42.6          5.11
1          5          156.7          18.8
1          6          189.3          22.72
1          7          214.1          25.69
1          8          296.7          35.6
1          9          326.8          39.22
1         10          650.1          78.01
1 - Inserir
```



# pesquisaporIDcidadeB():Algoritmo

Algoritmo: pesquisa\_por\_id\_cidade\_b

Objetivo: Permite pesquisar por uma cidade de chegada.

Constantes: FICHEIRO\_CIDADES (TEXTO 90) - Nome do ficheiro dos dados das cidades (Valor: cidades.txt)

Variáveis:

Entrada:

IDCidadeB (TEXTO 1)- Número da cidade (Números de 1-9)

Saída:

IDCidadeA (TEXTO 1)- Número da cidade (Números de 1-9)

IDCidadeB (TEXTO 1)- Número da cidade (Números de 1-9)

Distancia (TEXTO 30)- Distância entre as cidades ( $\geq 0, < 999$ )

Preço (TEXTO 30)- Preço ( $\geq 0, < 100$ )

Início:  
LER dados

ABRIR ficheiro  
cidades.txt

ESCREVER a cidade  
pesquisada

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# pesquisaporIDcidadeB():Python

```
def pesquisaporIDcidadeB():  
    idcidadeBProcurar = input("Diga a cidade de chegada a procurar [IDcidadeB]?")  
    fbrand = 'distancias.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidadeA= colunas[0]  
        idcidadeB= colunas[1]  
        distancia= colunas[2]  
        preco= colunas[3]  
        if (idcidadeB.find(idcidadeBProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(idcidadeA,idcidadeB,distancia,preco))  
    infile.close()
```



# pesquisaporIDcidadeB():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar

?6
Diga a cidade de chegada a procurar [IDcidadeB]?9
1          9          326.8          39.22
2          9          313.7          37.64
3          9           370           44.4
4          9          243.5          29.22
5          9          198.5          23.82
6          9          142.5          17.1
7          9           87.2          10.46
1 - Inserir
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço

0 - Voltar

?6
Diga a cidade de chegada a procurar [IDcidadeB]?7
1          7          214.1          25.69
2          7          215.4          25.85
3          7          298.3          35.8
4          7          165.3          19.84
5          7          105.2          12.62
6          7           28.9           3.47
1 - Inserir
```



# pesquisapordistancia():Algoritmo

Algoritmo: pesquisa\_por\_distancia

Objetivo: Permite pesquisar distâncias.

Constantes: FICHEIRO\_DISTANCIAS (TEXTO 90) - Nome do ficheiro dos dados das distancias (Valor: distancias.txt)

Variáveis:

Entrada:

Distância (TEXTO 30)- Distância entre as cidades (>=0,<999)

Saída:

IDCidadeA (TEXTO 1)- Número da cidade (Números de 1-9)

IDCidadeB (TEXTO 1)- Número da cidade (Números de 1-9)

Distancia (TEXTO 30)- Distância entre as cidades (>=0,<999)

Preço (TEXTO 30)- Preço (>=0,<100)Data: 2017-01-02

Início:  
LER dados

ABRIR ficheiro  
distancias.txt

ESCREVER a distancia  
pesquisada

FECHAR o ficheiro  
Fim.

Versão: 1.0

Obs:





# pesquisapordistancia():Python

```
def pesquisapordistancia():  
    distanciaProcurar = input("Diga a distancia a procurar?")  
    fbrand = 'distancias.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidadeA= colunas[0]  
        idcidadeB= colunas[1]  
        distancia= colunas[2]  
        preco= colunas[3]  
        if (distancia.find(distanciaProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(idcidadeA,idcidadeB,distancia,preco))  
    infile.close()
```



# pesquisapordistancia():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?7
```

```
Diga a distancia a procurar?100
```

4	6	100.3	12.04
6	8	100.2	12.02

```
1 - Inserir
```

```
1 - Inserir
```

```
2 - Alterar
```

```
3 - Eliminar
```

```
4 - Listar Todos
```

```
5 - Pesquisa por cidade de partida
```

```
6 - Pesquisa por cidade de chegada
```

```
7 - Pesquisa por distancia
```

```
8 - Pesquisa por Preço
```

```
0 - Voltar
```

```
?7
```

```
Diga a distancia a procurar?650
```

1	10	650.1	78.01
---	----	-------	-------



# pesquisaporpreço():Algoritmo

Algoritmo: pesquisa\_por\_preco

Objetivo: Permite pesquisar preços entre distâncias.

Constantes: FICHEIRO\_DISTANCIAS (TEXTO 90) - Nome do ficheiro dos dados das distancias (Valor: distancias.txt)

Variáveis:

Entrada:

Preço (TEXTO 30)- Preço ( $\geq 0, < 100$ )

Saída:

IDCidadeA (TEXTO 1)- Numero da cidade (Números de 1-9)

IDCidadeB (TEXTO 1)- Numero da cidade (Números de 1-9)

Distância (TEXTO 30)- distancia entre as cidades ( $\geq 0, < 999$ )

Preço (TEXTO 30)- Preço ( $\geq 0, < 100$ )

Início:  
LER dados

ABRIR ficheiro  
distancias.txt

ESCREVER o preço  
pesquisado

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# pesquisaporpreço():Python

```
def pesquisaporpreço():  
    precoProcurar = input("Diga o preço a procurar?")  
    fbrand = 'distancias.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        idcidadeA= colunas[0]  
        idcidadeB= colunas[1]  
        distancia= colunas[2]  
        preco= colunas[3]  
        if (preco.find(precoProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(idcidadeA,idcidadeB,distancia,preco))  
    infile.close()
```



# pesquisaporpreço():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
0 - Voltar
```

```
?8
Diga o preço a procurar?10
7          9          87.2          10.46
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todos
5 - Pesquisa por cidade de partida
6 - Pesquisa por cidade de chegada
7 - Pesquisa por distancia
8 - Pesquisa por Preço
0 - Voltar
```

```
?8
Diga o preço a procurar?25
1          7          214.1          25.69
2          7          215.4          25.85
3          6          214.2          25.7
```





# Gestão de passagens: Algoritmo

## Início:

ESCREVER(1 - Inserir)  
ESCREVER(2 - Alterar)  
ESCREVER(3 - Eliminar)  
ESCREVER(4 - Listar Todas)  
ESCREVER(5 - Pesquisa por cidade [IDCidade])  
ESCREVER(6 - Pesquisa por IDUtenteFK)  
ESCREVER(7 - Pesquisa por data)  
ESCREVER(8 - Pesquisa por entrada[0] ou saída[1])  
ESCREVER(9-Converter ficheiro .txt em .bin)  
ESCREVER()  
ESCREVER(0 - Voltar)  
ESCREVER()

## Início:

Se (0) então  
    Voltar atrás  
SeNão (1) então  
    inserirpassagem()  
SeNão (2) então  
    ReadAllLinesVectorAlterarPassagem()  
SeNão (3) então  
    ReadAllLinesVectorEliminarPassagem()  
SeNão (4) então  
    listatodaspassagens()  
SeNão (5)então  
    pesquisaporIDCidade()  
SeNão (6) então  
    pesquisaporIDUtenteFK()  
SeNão (7) então  
    pesquisapordata()  
SeNão (8) então  
    pesquisaporentradasaida()  
SeNão (9) então  
    PassagensTXT2BIN()  
SeNão:  
    ESCREVER(Escolha uma opção contida na lista apresentada)  
FimSe  
FimSe  
**Fim.**



# Gestão de passagens: Python

```
def gestaodepassagens ():
```

```
    while True:
```

```
        print("1 - Inserir")
```

```
        print("2 - Alterar")
```

```
        print("3 - Eliminar")
```

```
        print("4 - Listar Todas")
```

```
        print("5 - Pesquisa por cidade [IDCidade]")
```

```
        print("6 - Pesquisa por IDUtenteFK")
```

```
        print("7 - Pesquisa por data")
```

```
        print("8 - Pesquisa por entrada[escrever 0] ou saida[escrever 1]")
```

```
        print("9 - Converter ficheiro .txt em .bin")
```

```
        print()
```

```
        print("0 - Voltar")
```

```
        print()()
```

```
    op=input("?")
```

```
    if op=="0":
```

```
        break
```

```
    elif op=="1":
```

```
        inserirpassagem()
```

```
    elif op=="2":
```

```
        ReadAllLinesVectorAlterarPassagem()
```

```
    elif op=="3":
```

```
        ReadAllLinesVectorEliminarPassagem()
```

```
    elif op=="4":
```

```
        listatodaspassagens()
```

```
    elif op=="5":
```

```
        pesquisaporIDCidade()
```

```
    elif op=="6":
```

```
        pesquisaporIDUtenteFK()
```

```
    elif op=="7":
```

```
        pesquisapordata()
```

```
    elif op=="8":
```

```
        pesquisaporentradasaida()
```

```
    elif op=="9":
```

```
        PassagensTXT2BIN()
```

```
    else:
```

```
        print("Escolha uma opção contida na lista apresentada")
```



# Gestão de passagens: : Demonstração

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?5
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]
9 - Converter ficheiro .txt em .bin

0 - Voltar
```

```
1 - Dono
2 - Carros
3 - Cidades
4 - Distâncias
5 - Passagens

0 - Terminar

?5
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]
9 - Converter ficheiro .txt em .bin

0 - Voltar

?
```





# inserirpassagem():Algoritmo

Algoritmo: Inserir\_passagem

Objetivo: Permite inserir uma passagem no ficheiro de texto passagensNOVO.txt.

Constantes: FICHEIRO\_PASSAGENSNOVO (TEXTO 40) - Nome do ficheiro dos dados das passagens (Valor: passagensNOVO.txt)

Variáveis

Entrada:

IDCidade (TEXTO 1)- Numero da cidade	(Números de 1-9)
IDUtenteFK (TEXTO 40)- Dono do carro	(>=1;<=9999999999)
Data e hora (TEXTO 30)- data e hora	(>=0,<999)
Entrada ou saída (TEXTO 30)- entrada ou saída	(entrada/saída)

Saída:

Data: 2016-11-21

Versão: 1.0

Obs:

Início:  
LER dados

ABRIR ficheiro  
passagensNOVO.txt  
para escrita no fim

ESCREVER os dados no  
ficheiro

FECHAR o ficheiro  
Fim.



# inserirpassagem():Python

```
def inserirpassagem():  
    IDCidade = eval(input("IDCidade?"))  
    IDUtenteFK = input("IDUtenteFK ?")  
    Data = input("data e hora ?")  
    Entsai = input("Entrada-0 ou saida-1 ?")  
    ref_ficheiro = open("passagensNOVO.txt", "at")  
    print(IDCidade, ";", IDUtenteFK, ";", Data, ";", Entsai, ";", file=ref_ficheiro)  
    ref_ficheiro.close()
```



# inserirpassagem():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saida[escrever 1]

0 - Voltar

?1
IDCidade?9
IDUtenteFK ?414154
data e hora ?12:45
Entrada-0 ou saida-1 ?0
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saida[escrever 1]

0 - Voltar

?1
IDCidade?8
IDUtenteFK ?465145
data e hora ?8:30
Entrada-0 ou saida-1 ?1
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saida[escrever 1]

0 - Voltar
```



# ReadAllLinesVectorAlterarPassagem():Algoritmo

Algoritmo: alterar\_passagem

Objetivo: Permite alterar os dados de uma passagem escolha.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens (Valor: passagens.txt)

Variáveis:

Entrada:

IDCidade (TEXTO 1)- Numero da cidade (Números de 1-9)

IDUtenteFK (TEXTO 40)- Dono do carro ( $\geq 1; \leq 9999999999$ )

Data e hora (TEXTO 30)- data e hora ( $\geq 0, < 999$ )

Entrada ou saída (TEXTO 30)- entrada ou saída (entrada/saída)

Saída:

Início:  
Ler dados

ABRIR ficheiro  
passagens.txt

ESCREVER os dados no  
ficheiro passagens.txt

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# ReadAllLinesVectorAlterarPassagem():Python(pt1)

```
def ReadAllLinesVectorAlterarPassagem():
```

```
    fname = 'passagens.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    SearchIDUtenteFK = input("IDUtenteFK a alterar?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 4):
```

```
            continue
```

```
        IDCidade = colunas[0]
```

```
        IDUtenteFK = colunas[1]
```

```
        Data = colunas[2]
```

```
        EntradaSaida = colunas[3]
```

```
    if (SearchIDUtenteFK == IDUtenteFK):
```

```
        pos = i
```

```
        print ('\n' * 3)
```

```
        print ('IDUtenteFK encontrado na posição ', i)
```

```
        print ('\n' * 2)
```

```
        print ('IDCidade.....:', IDCidade)
```

```
        print ('IDUtenteFK.....:', IDUtenteFK)
```

```
        print ('Data.....:', Data)
```

```
        print ('Entrada-0 Saida-1.....:', EntradaSaida)
```

```
        op = input("Alterar ? (s/n) ?")
```



# ReadAllLinesVectorAlterarPassagem():Python(pt2)

```
if (op == 's'):
    print('Alterar ')
    IDCidade    = input("IDCidade ?")
    IDUtenteFK   = input("IDUtenteFK ?")
    Data        = input("Data (DD-MM-AAAA HH:MM:SS.sss) ?")
    EntradaSaida = eval(input("Entrada-0 Saida-1 ?"))
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
        else:
            print(IDCidade, IDUtenteFK, Data, EntradaSaida, sep=';', file=outfile)
    outfile.close();
    break;
```

```
if (pos == -1):
    print('Essa passagem não existe')
```



# ReadAllLinesVectorAlterarPassagem():Demonstração

```
Escolha uma opção contida na lista apresentada
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]

0 - Voltar

?2
Registos: 541531
IDUtenteFK a alterar?3952

IDUtenteFK encontrado na posição 21726

IDCidade..... : 4
IDUtenteFK..... : 3952
Data..... : 02-10-2004 18:41:05.503
Entrada-0 Saída-1..... : 0

Alterar ? (s/n) ?s
Alterar
IDCidade ?3
IDUtenteFK ?3569
Data (DD-MM-AAAA HH:MM:SS.sss) ?11-04-2016
Entrada-0 Saída-1 ?0
```

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]

0 - Voltar

?2
Registos: 541531
IDUtenteFK a alterar?2115

IDUtenteFK encontrado na posição 10119

IDCidade..... : 8
IDUtenteFK..... : 2115
Data..... : 04-07-2005 17:33:12.797
Entrada-0 Saída-1..... : 0

Alterar ? (s/n) ?s
Alterar
IDCidade ?6
IDUtenteFK ?1125
Data (DD-MM-AAAA HH:MM:SS.sss) ?03-04-2016
Entrada-0 Saída-1 ?1
```



# ReadAllLinesVectorEliminarPassagem():Algoritmo

Algoritmo: eliminar\_passagens

Objetivo: Permite eliminar os dados de uma passagem à escolha.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens (Valor: passagens.txt)

Variáveis:

Entrada:

Data e hora (TEXTO 30)- data e hora

( $\geq 0, < 999$ )

Saída:

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

Ler dados

ABRIR ficheiro  
passagens.txt

ELIMINAR os dados do  
ficheiro passagens.txt  
referentes à passagem  
pesquisada

FECHAR o ficheiro  
Fim.





# ReadAllLinesVectorEliminarPassagem():Python(pt1)

```
def ReadAllLinesVectorEliminarPassagem():
```

```
    fname = 'passagens.txt'
```

```
    infile = open(fname, "r")
```

```
    lines = infile.readlines()
```

```
    infile.close()
```

```
    print("Registos: ", len(lines))
```

```
    searchData = input("Data ?")
```

```
    pos = -1
```

```
    for i in range(len(lines)):
```

```
        colunas = lines[i].split(';')
```

```
        if (len(colunas) < 4):
```

```
            continue
```

```
        IDCidade = colunas[0]
```

```
        IDUtenteFK = colunas[1]
```

```
        Data = colunas[2]
```

```
        EntradaSaida = colunas[3]
```

```
    if (SearchData == Data):
```

```
        pos = i
```

```
        print ('\n' * 3)
```

```
        print ('Data encontrado na posição ', i)
```

```
        print ('\n' * 2)
```

```
        print ('IDCidade.....:', IDCidade)
```

```
        print ('IDUtenteFK....:', IDUtenteFK)
```

```
        print ('Data.....:', Data)
```

```
        print ('EntradaSaida..:', EntradaSaida)
```

```
        op = input("Eliminar ? (s/n) ?")
```



# ReadAllLinesVectorEliminarPassagem():Python(pt2)

```
if (op == 's'):
    print('Eliminando ... ')
    outfile = open(fname, "w")
    for i in range(len(lines)):
        if (i != pos):
            line = lines[i]
            line[0:len(line)-1]
            print(line, file=outfile, end="")
    outfile.close();
    break;
if (pos == -1):
    print('Essa passagem não existe')
```



# ReadAllLinesVectorEliminarPassagem():Demonstração

```
Escolha uma opção contida na lista apresentada
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]

0 - Voltar

?3
Registos: 541531
Data ?11-04-2016

Data encontrado na posição 21726

IDCidade..... : 3
IDUtenteFK.... : 3569
Data..... : 11-04-2016
EntradaSaída.. : 0

Eliminar ? (s/n) ?s
Eliminando ...
```



# listatodaspassagens():Algoritmo

Algoritmo: listar\_todas\_passagens

Objetivo: Permite listar todas as passagens.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens  
(Valor: passagens.txt)

Variáveis:

Entrada:

Saída:

IDCidade(TEXTO 1)- Numero da cidade	(Números de 1-9)
IDUtenteFK(TEXTO 40)- Dono do carro	(>=1,<=999999999)
Data e hora (TEXTO 30)- data e hora	(>=0,<999)
Entrada ou saída(TEXTO 30)- entrada ou saída	(entrada/saída)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:  
ABRIR ficheiro  
passagens.txt

ESCREVER os dados  
do ficheiro  
passagens.txt

FECHAR o ficheiro  
Fim.

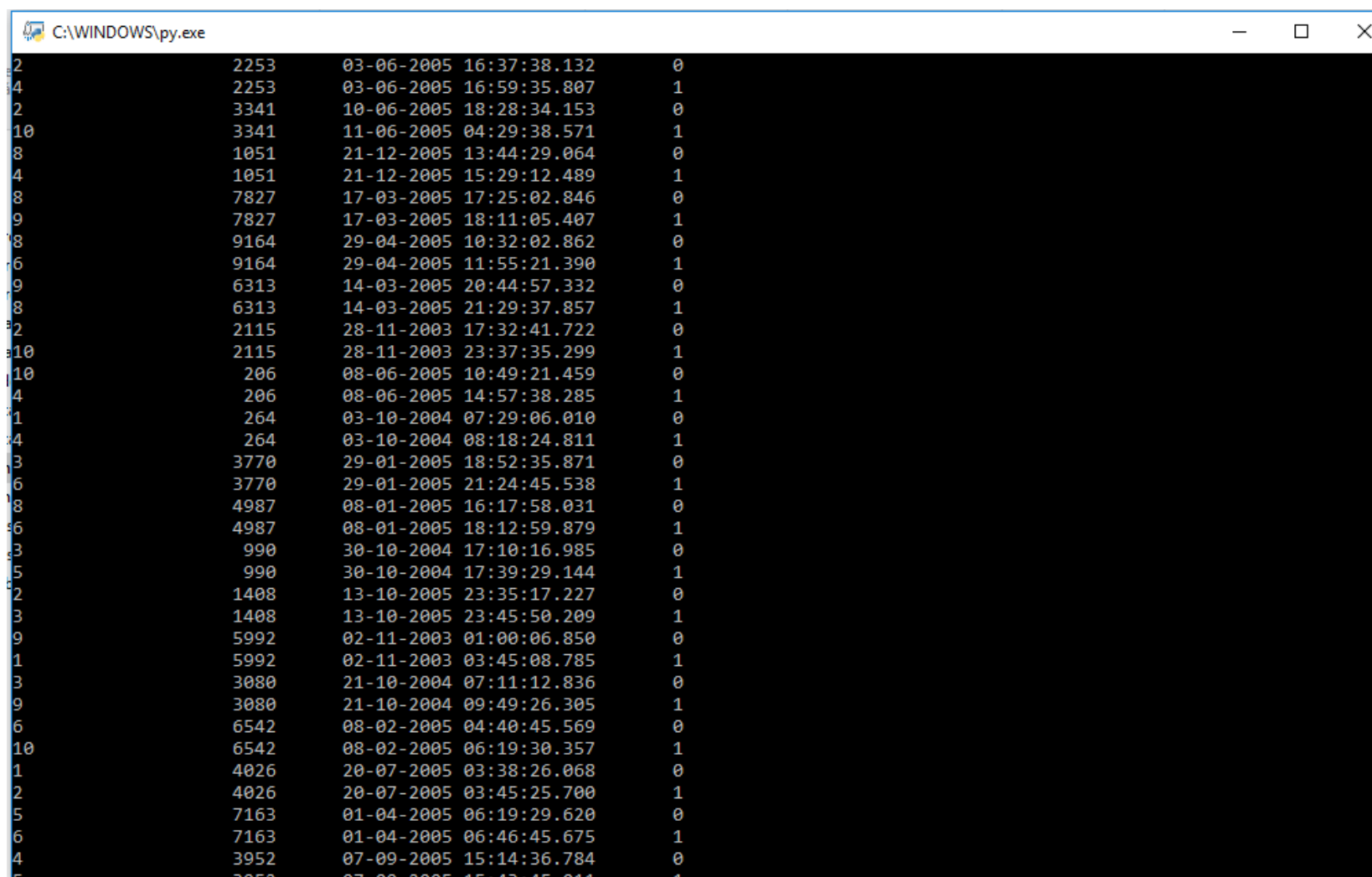


# listatodaspassagens():Python

```
def listatodaspassagens():  
    fbrand = 'passagens.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDCidade = colunas[0]  
        IDUtenteFK = colunas[1]  
        Data = colunas[2]  
        Entsai = colunas[3]  
        print ("{0:15}{1:^15}{2:^15}{3:^15}".format(IDCidade,IDUtenteFK,Data,Entsai))  
    infile.close()
```



# listatodaspassagens():Demonstração



2	2253	03-06-2005	16:37:38.132	0
4	2253	03-06-2005	16:59:35.807	1
2	3341	10-06-2005	18:28:34.153	0
10	3341	11-06-2005	04:29:38.571	1
8	1051	21-12-2005	13:44:29.064	0
4	1051	21-12-2005	15:29:12.489	1
8	7827	17-03-2005	17:25:02.846	0
9	7827	17-03-2005	18:11:05.407	1
8	9164	29-04-2005	10:32:02.862	0
6	9164	29-04-2005	11:55:21.390	1
9	6313	14-03-2005	20:44:57.332	0
8	6313	14-03-2005	21:29:37.857	1
2	2115	28-11-2003	17:32:41.722	0
10	2115	28-11-2003	23:37:35.299	1
10	206	08-06-2005	10:49:21.459	0
4	206	08-06-2005	14:57:38.285	1
1	264	03-10-2004	07:29:06.010	0
4	264	03-10-2004	08:18:24.811	1
3	3770	29-01-2005	18:52:35.871	0
6	3770	29-01-2005	21:24:45.538	1
8	4987	08-01-2005	16:17:58.031	0
6	4987	08-01-2005	18:12:59.879	1
3	990	30-10-2004	17:10:16.985	0
5	990	30-10-2004	17:39:29.144	1
2	1408	13-10-2005	23:35:17.227	0
3	1408	13-10-2005	23:45:50.209	1
9	5992	02-11-2003	01:00:06.850	0
1	5992	02-11-2003	03:45:08.785	1
3	3080	21-10-2004	07:11:12.836	0
9	3080	21-10-2004	09:49:26.305	1
6	6542	08-02-2005	04:40:45.569	0
10	6542	08-02-2005	06:19:30.357	1
1	4026	20-07-2005	03:38:26.068	0
2	4026	20-07-2005	03:45:25.700	1
5	7163	01-04-2005	06:19:29.620	0
6	7163	01-04-2005	06:46:45.675	1
4	3952	07-09-2005	15:14:36.784	0
5	3952	07-09-2005	15:43:45.011	1



# pesquisaporIDCidade():Algoritmo

Algoritmo: pesquisa\_por\_id\_cidade

Objetivo: Permite pesquisar por passagens numa cidade.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens  
(Valor: passagens.txt)

Variáveis:

Entrada:

IDCidade (TEXTO 1)- Numero da cidade (Números de 1-9)

Saída:

IDCidade (TEXTO 1)- Numero da cidade (Números de 1-9)

IDUtenteFK(TEXTO 40)- Dono do carro (>=1,<=999999999)

Data e hora (TEXTO 30)- data e hora (>=0,<999)

Entrada ou saída(TEXTO 30)- entrada ou saída (entrada/saída)

Início:  
LER dados

ABRIR ficheiro  
passagens.txt

ESCREVER as  
passagens pesquisadas

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# pesquisaporIDCidade():Python

```
def pesquisaporIDCidade():  
    IDCidadeProcurar = input("Diga a cidade a procurar [IDcidade]?")  
    fbrand = 'passagens.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDCidade= colunas[0]  
        IDUtenteFK= colunas[1]  
        Data= colunas[2]  
        Entsai= colunas[3]  
        if (IDCidade.find(IDCidadeProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(IDCidade,IDUtenteFK,Data,Entsai))  
    infile.close()
```





# pesquisaporIDCidade():Demonstração

6340	16-02-2005	16:23:07.620	0
7180	06-02-2005	16:54:48.927	1
3699	11-05-2004	08:03:39.301	1
5532	20-01-2006	08:39:14.850	1
4824	26-04-2004	03:12:45.953	1
1881	03-02-2005	17:20:14.474	0
2102	05-04-2004	13:58:00.120	0
5798	02-06-2005	02:05:36.225	0
7118	08-07-2004	11:19:10.301	0
3646	05-02-2005	21:20:26.691	0
1802	10-09-2005	15:43:45.264	0
541	14-12-2003	12:09:15.041	0
6317	04-04-2005	03:35:42.430	0
3466	09-12-2004	13:49:03.691	1
2084	22-08-2004	15:55:02.345	1
1889	20-07-2005	19:40:47.166	1
2851	15-11-2003	05:45:18.726	0
1435	05-09-2005	21:13:34.377	1
2378	10-01-2005	20:38:12.967	0
7007	25-11-2005	20:12:46.970	0
4871	21-05-2005	18:47:36.458	0
9009	08-06-2004	12:01:55.614	0
3847	25-11-2004	03:50:37.580	1
4408	27-08-2005	12:42:28.296	1
442	03-02-2005	18:04:03.364	0
3635	26-07-2004	18:24:59.502	0
4420	22-03-2005	10:53:41.259	1
1362	19-04-2005	21:27:25.957	0
9014	04-01-2005	09:06:35.049	1
4072	17-07-2004	11:45:39.614	1
6171	19-01-2005	01:43:57.827	0
1773	12-01-2005	12:23:32.543	0
5697	14-08-2005	16:10:56.158	1
7996	12-12-2005	16:07:46.936	1
294	11-12-2004	20:00:14.969	0
3725	20-12-2004	23:28:29.650	1
1098	21-04-2005	16:42:25.255	1
753	04-06-2005	15:00:14.708	0



# pesquisaporIDUtenteFK():Algoritmo

Algoritmo: pesquisa\_por\_id\_utente

Objetivo: Permite pesquisar por utente.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens (Valor: passagens.txt)

Variáveis:

Entrada:

IDUtenteFK(TEXTO 40)- Dono do carro (Números de 1-9)

Saída:

IDCidade (TEXTO 1)- Numero da cidade (Números de 1-9)

IDUtenteFK(TEXTO 40)- Dono do carro ( $\geq 1, \leq 999999999$ )

Data e hora (TEXTO 30)- data e hora ( $\geq 0, < 999$ )

Entrada ou saída(TEXTO 30)- entrada ou saída (entrada/saída)

Início:  
LER dados

ABRIR ficheiro  
passagens.txt

ESCREVER o utente  
pesquisado

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:



# pesquisaporIDUtenteFK():Python

```
def pesquisaporIDUtenteFK():
    IDUtenteFKProcurar = input("Diga o IDUtenteFK a procurar?")
    fbrand = 'passagens.txt'
    infile = open(fbrand, "r")
    for line in infile:
        line = line[0:len(line)-1]
        colunas = line.split(';')
        IDCidade= colunas[0]
        IDUtenteFK= colunas[1]
        Data= colunas[2]
        Entsai= colunas[3]
        if (IDUtenteFK.find(IDUtenteFKProcurar)>=0):
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(IDCidade,IDUtenteFK,Data,Entsai))
    infile.close()
```



# pesquisaporIDUtenteFK():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]

0 - Voltar

?6
Diga o IDUtenteFK a procurar?3646
7          3646      14-12-2004 15:55:01.015      0
9          3646      14-12-2004 16:49:16.078      1
4          3646      26-12-2004 17:50:44.550      0
7          3646      26-12-2004 20:15:49.463      1
10         3646      16-09-2005 22:46:40.773      0
4          3646      17-09-2005 02:18:32.771      1
6          3646      10-04-2005 12:50:34.366      0
5          3646      10-04-2005 13:35:56.641      1
2          3646      07-04-2005 06:11:21.473      0
8          3646      08-04-2005 04:16:47.928      1
7          3646      14-01-2005 19:31:19.461      0
6          3646      14-01-2005 19:43:47.832      1
3          3646      13-10-2005 21:35:21.708      0
10         3646      14-10-2005 01:43:49.469      1
1          3646      11-07-2004 09:11:38.369      0
6          3646      11-07-2004 10:54:02.434      1
1          3646      19-03-2005 21:29:24.905      0
3          3646      19-03-2005 21:48:24.138      1
9          3646      21-11-2004 23:32:57.022      0
5          3646      21-11-2004 16:22:45.237      1
7          3646      10-01-2004 23:44:57.653      0
4          3646      14-01-2005 04:02:24.878      1
```



# pesquisapordata():Algoritmo

Algoritmo: pesquisa\_por\_data\_hora

Objetivo: Permite pesquisar a data e a hora de uma passagem.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens  
(Valor: passagens.txt)

Variáveis:

Entrada:

Data e hora (TEXTO 30)- data e hora ( $\geq 0, < 999$ )

Saída:

IDCidade (TEXTO 1)- Numero da cidade (Números de 1-9)

IDUtenteFK(TEXTO 40)- Dono do carro ( $\geq 1, \leq 9999999999$ )

Data e hora (TEXTO 30)- data e hora ( $\geq 0, < 999$ )

Entrada ou saída(TEXTO 30)- entrada ou saída (entrada/saída)

Data: 2017-01-02

Versão: 1.0

Obs:

Início:

LER dados

ABRIR ficheiro  
passagens.txt

ESCREVER a  
passagem à data e  
a hora pesquisada

FECHAR o ficheiro  
Fim.



# pesquisapordata():Pyhton

```
def pesquisapordata():  
    DataProcurar = input("Diga a data e hora a procurar?")  
    fbrand = 'passagens.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDCidade= colunas[0]  
        IDUtenteFK= colunas[1]  
        Data= colunas[2]  
        Entsai= colunas[3]  
        if (Data.find(DataProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(IDCidade,IDUtenteFK,Data,Entsai))  
    infile.close()
```



# pesquisapordata():Demonstração

```
Selecionar C:\WINDOWS\py.exe
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saida[escrever 1]
0 - Voltar

?7
Diga a data e hora a procurar?26-03-2004
2          5503      26-03-2004 23:03:10.726      0
4          5953      26-03-2004 10:34:06.287      0
6          5953      26-03-2004 11:14:15.672      1
5          5418      26-03-2004 01:43:20.789      0
4          5418      26-03-2004 02:45:23.364      1
9          2235      26-03-2004 01:35:09.608      0
1          2235      26-03-2004 06:06:34.036      1
7          1512      26-03-2004 05:37:18.699      0
8          1512      26-03-2004 06:14:51.265      1
5          6293      26-03-2004 12:56:16.110      0
2          6293      26-03-2004 14:00:27.069      1
4          9967      26-03-2004 01:37:11.550      0
1          9967      26-03-2004 02:07:49.346      1
10         8136      26-03-2004 01:32:17.382      1
10         2131      26-03-2004 01:08:02.261      1
6          6804      26-03-2004 00:30:19.781      1
5          1593      26-03-2004 06:04:34.377      0
3          1593      26-03-2004 06:55:16.852      1
8          4332      26-03-2004 22:19:34.815      0
3          4881      26-03-2004 11:46:32.168      0
9          4881      26-03-2004 16:39:25.818      1
1          5354      26-03-2004 18:33:34.665      0
3          5354      26-03-2004 19:01:26.165      1
5          3256      26-03-2004 00:02:56.307      1
```

```
*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saida[escrever 1]
0 - Voltar

?7
Diga a data e hora a procurar?14-01-2005
5          9687      14-01-2005 05:27:05.594      0
6          9687      14-01-2005 06:25:11.071      1
3          8097      14-01-2005 05:37:44.184      0
8          8097      14-01-2005 08:37:36.216      1
7          361       14-01-2005 22:10:24.241      0
2          437       14-01-2005 12:22:34.682      0
4          437       14-01-2005 12:46:45.623      1
9          6811      14-01-2005 09:41:02.208      0
2          6811      14-01-2005 12:25:39.608      1
8          3136      14-01-2005 03:02:10.582      0
1          3136      14-01-2005 06:31:44.180      1
8          2308      14-01-2005 09:03:48.422      0
10         2308      14-01-2005 11:08:15.495      1
5          3588      14-01-2005 11:30:27.030      0
7          3588      14-01-2005 12:11:17.369      1
6          7087      14-01-2005 02:13:42.302      0
5          7087      14-01-2005 03:31:51.458      1
10         3232      14-01-2005 20:21:05.278      0
6          3232      14-01-2005 21:02:48.279      1
10         1758      14-01-2005 21:41:04.303      0
9          1758      14-01-2005 22:59:11.469      1
8          5023      14-01-2005 15:34:21.898      0
1          5023      14-01-2005 18:18:01.264      1
4          9124      14-01-2005 00:14:11.019      0
1          9124      14-01-2005 00:28:13.746      1
9          946       14-01-2005 22:13:34.856      0
7          2558      14-01-2005 14:51:30.217      0

Ln: 48 Col: 0
```



# pesquisaporentradadasaida():Algoritmo

Algoritmo: pesquisa\_por\_entrada\_saida

Objetivo: Permite pesquisar se uma certa passagem foi entrada ou saída.

Constantes: FICHEIRO\_PASSAGENS (TEXTO 90) - Nome do ficheiro dos dados das passagens (Valor: passagens.txt)

Variáveis:

Entrada:

Entrada ou saída(TEXTO 30) - entrada ou saída (entrada/saída)

Saída:

IDCidade (TEXTO 1)- Numero da cidade (Números de 1-9)

IDUtenteFK(TEXTO 40)- Dono do carro ( $\geq 1, \leq 999999999$ )

Data e hora (TEXTO 30)- data e hora ( $\geq 0, < 999$ )

Entrada ou saída(TEXTO 30)- entrada ou saída (entrada/saída)

Início:  
LER dados

ABRIR ficheiro  
passagens.txt

ESCREVER se foi  
entrada ou saída

FECHAR o ficheiro  
Fim.

Data: 2017-01-02

Versão: 1.0

Obs:





# pesquisaporentradasaida():Python

```
def pesquisaporentradasaida():  
    EntsaiProcurar = input("O que pretende procurar? [entrada -> 0 ; saida -> 1]")  
    fbrand = 'passagens.txt'  
    infile = open(fbrand, "r")  
    for line in infile:  
        line = line[0:len(line)-1]  
        colunas = line.split(';')  
        IDCidade= colunas[0]  
        IDUtenteFK= colunas[1]  
        Data= colunas[2]  
        Entsai= colunas[3]  
        if (Entsai.find(EntsaiProcurar)>=0):  
            print ("{0:15}{1:^15}{2:^15}{3:^15}".format(IDCidade,IDUtenteFK,Data,Entsai))  
    infile.close()
```



# pesquisaporentradadasaida():Demosntração

C:\WINDOWS\py.exe

2	1120	14-02-2005	23:31:14.265	0
9	6007	04-05-2005	10:48:06.858	0
4	8995	20-07-2005	10:46:39.776	0
8	4655	01-06-2005	09:08:28.145	0
8	1839	30-10-2004	23:18:54.644	0
2	4554	22-03-2005	21:38:21.985	0
9	21	17-01-2004	02:57:28.409	0
4	9740	06-05-2004	11:39:45.090	0
5	7334	25-07-2004	05:39:30.615	0
2	7672	12-07-2005	23:56:48.629	0
10	1116	30-12-2004	00:41:25.494	0
1	9409	08-12-2003	08:02:56.410	0
2	3627	07-02-2006	06:19:46.679	0
10	9087	27-06-2004	20:46:19.472	0
7	8514	29-06-2005	06:13:01.516	0
10	4979	16-08-2004	11:08:41.783	0
8	7753	24-12-2003	06:39:42.360	0
4	7268	04-09-2003	22:49:32.002	0
1	1722	29-12-2003	17:10:02.406	0
9	3971	30-06-2004	02:10:14.262	0
4	935	09-07-2005	05:19:31.834	0
4	1904	15-06-2004	05:30:25.687	0
5	1906	11-09-2005	04:51:30.132	0
4	6853	27-07-2004	02:07:16.996	0
9	280	17-09-2005	11:43:09.899	0
8	5675	10-01-2005	15:50:14.475	0
9	557	21-09-2004	15:21:09.157	0
9	2335	04-06-2005	15:18:56.212	0
4	9317	23-04-2005	11:11:18.455	0
10	5048	26-12-2005	07:13:19.052	0
6	6730	11-12-2004	06:37:46.545	0
6	2575	15-12-2004	02:17:59.095	0
3	8941	05-06-2005	13:27:18.727	0
7	3546	23-07-2005	10:48:27.276	0
2	8414	28-04-2004	18:21:08.480	0
1	3823	10-10-2004	15:07:11.698	0
4	8017	12-06-2004	04:23:03.201	0
3	9858	09-04-2004	12:41:18.917	0
3	6851	31-10-2005	21:52:35.705	0
7	6554	29-08-2004	08:51:51.817	0
10	6823	29-11-2004	16:34:28.571	0
4	3968	05-01-2005	03:16:18.213	0
10	8736	11-05-2004	00:29:36.500	0
1	5919	27-04-2004	08:10:46.826	0
5	6495	16-01-2005	20:26:03.424	0
3	1071	27-07-2005	13:21:09.472	0
1	5194	12-10-2004	20:30:09.036	0
9	2269	01-02-2004	12:28:04.492	0
2	9621	29-03-2005	06:54:14.662	0
2	3071	09-02-2006	00:43:21.107	0
6	6036	14-09-2004	07:30:00.660	0
8	8545	29-09-2004	08:49:01.485	0
6	4973	27-03-2004	16:28:48.824	0
4	7340	10-03-2005	18:26:01.967	0
10	5624	20-04-2005	11:58:57.632	0
6	2881	08-06-2005	04:06:47.357	0
3	1402	13-06-2005	06:02:49.504	0
5	4546	20-04-2005	07:57:34.884	0
3	5393	17-12-2005	01:14:55.302	0
7	7612	11-09-2005	17:12:00.507	0
1	8184	19-09-2004	09:22:13.768	0
10	2467	17-01-2005	20:29:37.591	0
10	697	16-12-2005	20:21:59.371	0
7	0000	05-12-2004	12:05:10.004	0

C:\WINDOWS\py.exe

8	3137	14-12-2004	10:34:42.716	1
4	1120	15-02-2005	00:55:42.122	1
10	6007	04-05-2005	12:59:16.955	1
6	8995	20-07-2005	11:19:09.536	1
5	4655	01-06-2005	09:45:39.875	1
7	1839	30-10-2004	23:54:16.028	1
5	4554	22-03-2005	23:05:17.645	1
2	21	17-01-2004	05:57:27.859	1
7	9740	06-05-2004	13:01:04.135	1
4	7334	25-07-2004	06:10:01.379	1
4	7672	13-07-2005	00:20:36.907	1
4	1116	30-12-2004	17:43:32.352	1
4	9409	08-12-2003	08:16:15.839	1
10	3627	07-02-2006	11:34:45.406	1
4	9087	28-06-2004	00:27:11.928	1
1	8514	29-06-2005	21:01:56.489	1
2	4979	16-08-2004	17:57:57.703	1
10	7753	24-12-2003	10:13:39.895	1
1	7268	04-09-2003	23:14:59.719	1
8	1722	29-12-2003	19:31:27.037	1
7	3971	30-06-2004	08:00:07.109	1
7	935	09-07-2005	07:10:41.053	1
9	1904	15-06-2004	14:53:18.715	1
2	1906	11-09-2005	06:19:44.133	1
3	6853	27-07-2004	03:06:33.594	1
7	280	17-09-2005	12:39:03.216	1
6	5675	10-01-2005	19:16:01.508	1
4	557	21-09-2004	17:45:05.278	1
7	2335	04-06-2005	17:17:56.119	1
1	9317	23-04-2005	11:32:50.767	1
9	5048	26-12-2005	11:11:48.930	1
4	6730	11-12-2004	07:15:43.234	1
9	2575	15-12-2004	03:30:22.907	1
10	8941	05-06-2005	21:15:40.985	1
1	3546	23-07-2005	17:40:35.671	1
10	8414	29-04-2004	06:13:59.494	1
2	3823	10-10-2004	15:19:09.924	1
7	8017	12-06-2004	05:41:34.867	1
9	9858	09-04-2004	16:58:51.121	1
4	6851	31-10-2005	22:28:07.103	1
9	6554	29-08-2004	09:52:31.080	1
1	6823	29-11-2004	22:51:33.472	1
6	3968	05-01-2005	04:08:03.718	1
1	8736	11-05-2004	05:49:06.117	1
8	5919	27-04-2004	09:45:06.383	1
4	6495	16-01-2005	20:54:35.782	1
6	1071	27-07-2005	15:02:21.805	1
7	5194	12-10-2004	22:39:19.170	1
6	2269	01-02-2004	14:26:46.325	1
9	9621	29-03-2005	10:19:16.708	1
5	3071	09-02-2006	03:23:09.175	1
10	6036	14-09-2004	08:55:23.302	1
10	8545	29-09-2004	11:05:37.641	1
1	4973	27-03-2004	18:20:54.022	1
9	7340	10-03-2005	21:00:55.726	1
4	5624	20-04-2005	16:56:06.532	1
9	2881	08-06-2005	05:13:54.974	1
7	1402	13-06-2005	08:05:34.018	1
9	4546	20-04-2005	09:11:15.662	1
10	5393	17-12-2005	09:52:29.697	1
8	7612	11-09-2005	18:12:45.817	1
8	8184	19-09-2004	10:51:04.028	1
7	2467	17-01-2005	22:42:28.035	1

# PassagensTXT2BIN():Python

```
def PassagensTXT2BIN():
```

```
    # IDCidade;IDUtenteFK;Data;EntradaSaida
```

```
    # 4;1006;25-06-2004 07:43:52.803;0
```

```
    # 7;1006;25-06-2004 09:31:24.440;1
```

```
    import struct
```

```
    import datetime
```

```
    import time
```

```
    passagemFormato = struct.Struct('iifi');
```

```
    f_texto = open('passagens.txt', "r")
```

```
    f_bin = open("passagens.bin", "wb")
```

```
    lines = f_texto.readlines();
```

```
    i = 0
```

```
    print ('Início de conversão')
```

```
    start = start = time.time()
```

```
    for i in range(1, len(lines)):
```

```
        #for i in range(2, 10000):
```

```
            line = lines[i]
```

```
            line = line[0:len(line)-1]
```

```
            colunas = line.split(';')
```

```
            IDCidade = eval(colunas[0])
```

```
            IDUtenteFK = eval(colunas[1])
```

```
            EntradaSaida = eval(colunas[3])
```

```
            #print(i,IDCidade, IDUtenteFK, colunas[2], EntradaSaida)
```

```
            Data = datetime.datetime.strptime(colunas[2], "%d-%m-%Y  
            %H:%M:%S.%f")
```

```
            # float
```

```
            Data = time.mktime(Data.timetuple())
```

```
            passagemBinario = passagemFormato.pack(IDCidade, IDUtenteFK, Data,  
            EntradaSaida)
```

```
            f_bin.write(passagemBinario)
```

```
            i = i + 1
```

```
            if ((i % 1000)==0):
```

```
                print (i, sep=' ', end="")
```

```
    print ('Fim de conversão')
```

```
    f_texto.close()
```

```
    f_bin.close()
```

```
    end = time.time()
```

```
    print('Tempo de conversão (segundos)', end - start)
```

# PassagensTXT2BIN():Demonstração

```
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]
9 - Converter ficheiro .txt em .bin

0 - Voltar

?9
Início de conversão
100020003000400050006000700080009000100001100012000130001400015
027000280002900030000310003200033000340003500036000370003800039
051000520005300054000550005600057000580005900060000610006200063
4190004200042100042200042300042400042500042600042700042800
43900044000044100044200044300044400044500044600044700044800
45900046000046100046200046300046400046500046600046700046800
47900048000048100048200048300048400048500048600048700048800
49900050000050100050200050300050400050500050600050700050800
51900052000052100052200052300052400052500052600052700052800
539000540000541000Fim de conversão
Tempo de conversão (segundos) 24.00922703742981
1 - Inserir
```

```
*Python 3.6.0 Shell*
File Edit Shell Debug Options Window Help
1 - Inserir
2 - Alterar
3 - Eliminar
4 - Listar Todas
5 - Pesquisa por cidade [IDCidade]
6 - Pesquisa por IDUtenteFK
7 - Pesquisa por data
8 - Pesquisa por entrada[escrever 0] ou saída[escrever 1]
9 - Converter ficheiro .txt em .bin

0 - Voltar

?9
Início de conversão
100020003000400050006000700080009000100001100012000130001400
019000200002100022000230002400025000260002700028000290003000
035000360003700038000390004000041000420004300044000450004600
051000520005300054000550005600057000580005900060000610006200
067000680006900070000710007200073000740007500076000770007800
083000840008500086000870008800089000900009100092000930009400
004460004470004480004490004500004510004520004530004540
459000460000461000462000463000464000465000466000467000
200047300047400047500047600047700047800047900048000048
004860004870004880004890004900004910004920004930004940
499000500000501000502000503000504000505000506000507000
200051300051400051500051600051700051800051900052000052
005260005270005280005290005300005310005320005330005340
539000540000541000Fim de conversão
Tempo de conversão (segundos) 35.5285108089447
1 - Inserir
```



# Conclusão:

Através do programa desenvolvido foi possível usar os ficheiros (( donos2.txt (10,000)),(carros.txt (10,000) )(cidades.txt (10))(distancias.txt (45))(passagens.txt (3,000,000) ) ) na construção de diversos menus de modo a facilitar a organização, estrutura, consulta e tratamento desses mesmos dados.