Curso:
UpSkill – Digital Skills and Jobs

Unidade de Formação:
Programação
Formador:
Paulo Jorge Costa Nunes

Duração (horas):
96 Sala de Aula (SA)
96 Sessão Sincronia (SS)

2024-01-23-SS 2024-01-23-SA
2024-01-25-SS 2024-01-26-SA
2024-01-30-SS 2024-01-30-SA

4. Coleções: Dicionários (Listas, Tuplas, Sets)
12. Manipulação de Ficheiros Comma Separated Values (CSV):
leitura para dicionário e escrita a partir de dicionário.

Mini-Projeto: Concurso de ...
Parte-I: Programa para gerar perguntas e respostas sobre capitais de países.
Parte-II: Programa para efetuar as perguntas, guardar as respostas e calcular a pontuação.
Parte-III: Adicionar à Parte-II voz Text to Speech (TTS) nas perguntas, dicas, tempo, etc.

22 de janeiro de 2024

Lista de abreviações e símbolos

API	Application Program Interface	2
CSV	Comma Separated Values	1
JSON	JavaScript Object Notion	2
SOLID	Single responsibility - Open–closed - Liskov substitution - Interface segregation - Dependency inversion	2
TTS	Text to Speech	1

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Listagens

Capítulo 1

Plano

	Capítulo	Horas
1	Variáveis e Tipos de Dados	3
2	Estruturas Lógicas e Condicionais	3
3	Estruturas de Decisão e Repetição	6
4	Coleções (Listas, Tuplas, Sets, etc)	15
5	Funções e Expressões Lambdas	6
6	Tratamento de Erros	6
7	Módulos	3
8	Iteradores e Geradores	3
9	Decorators	3
10	Entrada e Saída de Dados (validação, regex, etc)	9
11	Orientação a Objetos (Herança e Polimorfismo) Conceitos Single responsibility - Open–closed - Liskov substitution - Interface segregation - Dependency inversion (SOLID) em contexto de Python	12
12	Manipulação de Ficheiros Comma Separated Values (CSV) e JavaScript Object Notion (JSON)	9
13	Ligação a Bases de Dados	9
14	Ligação a Web Application Program Interfaces (APIs) (**??** - **??**)	9

Tabela 1.1: Plano de aulas

API: geonames https://public.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-500/api/?disjunctive.country&refine.timezone=Europe%2FParis

dump: https://download.geonames.org/export/dump

Filter, map, slice ———— »> [a[slice(0,3)] for a in A] [[1, 2, 3], [1, 2, 3], [1, 2, 3]] Or:

»> [list(filter(lambda x: x<=3, a)) for a in A] [[1, 2, 3], [1, 2, 3], [1, 2, 3]]

Working Days API for Developers v1.2|v1.3 API: https://api.workingdays.org/1.3/api-documentation https://www.dias-uteis.pt/

Capítulo 2

Coleções (Listas, Tuplas, Sets, Dictionaries)

2.1 Dictionary - Dicionários

https://www.programiz.com/python-programming/dictionary

Em Python, um dicionário é uma coleção que nos permite armazenar dados em pares chave-valor. Criamos dicionários colocando pares chave:valor entre chaves `{}`, separados por vírgulas. [1]. Exemplo:

```
significado_palavras = {
  "a": "A forma a pode ser [feminino singular de o], [abreviatura], [adjectivo de dois géneros], [
    artigo definido], [nome masculino], [prefixo], [preposição], [pronome demonstrativo], [pronome
    pessoal feminino] ou [símbolo]",
  "copo": "Recipiente cilíndrico e sem asas, usado para beber"
}
```

Listagem 2.1: Exemplo de dicionário: palavras do dicionário da Língua Portuguesa.

```
cesta_frutas = {"maça": 60, "uva": 20, "pera": 35, "abacaxi": 25, "laranja": 55} 55}
```

Listagem 2.2: Exemplo de dicionário: cesta de frutas.

Ao contrário das listas (sequências), que são indexadas por um intervalo de números, os dicionários são indexados por chaves, que podem ser de qualquer tipo imutável; strings e números sempre podem ser chaves.

É melhor pensar em um dicionário como um conjunto de pares `chave:valor`, com a exigência de que as chaves sejam exclusivas (dentro de um dicionário). Um par de chavetas cria um dicionário vazio: `{}`. Colocar uma lista separada por vírgulas de pares `chave:valor`, entre colchetes adiciona pares chave:valor iniciais ao dicionário; esta também é a forma como os dicionários são escritos na saída.

As principais operações em um dicionário são:

1. Armazenar um valor com alguma chave e extrair o valor fornecido pela chave: `dic[chave] = valor`.
2. Excluir um par `chave:valor` com `del`: `del dic["profissão"]` ou `dic.pop("idade")`.
3. Modificar o valor de uma chave que já está em uso, o valor antigo associado a essa chave será esquecido.
4. Verificar se uma chave existe: `chave in dic.kyes()`.
5. Obter o valor de uma chave com `dic[chave]`. A extração de uma chave inexistente dá.
6. Obter todos os valores: `dic.values()`.
7. Obter todas as chaves: `dic.kyes()`.
8. Obter todos os pares de chave:valor: `dic.items()`.

Executar `list(d)` num dicionário retorna uma lista de todas as chaves usadas no dicionário, em ordem de inserção (se você quiser classificá-lo, basta usar `sorted(d)`). Para verificar se uma única chave está no dicionário, use a palavra-chave in.

2.2 Criar Dicionário Vazio e Adicionar Itens

```
# https://dicionario.priberam.org/comboio.
dic = {}
dic["carro"] = "Veículo de rodas para transporte de pessoas ou mercadorias"
dic["comboio"] = "Conjunto de vagões ou carruagens engatadas umas nas outras e puxadas por uma
    locomotiva"
dic["mota"] = "Veículo de duas rodas accionado por um motor"
print (dic)
print (dic["comboio"])
# {'carro': 'Veículo de rodas para transporte de pessoas ou mercadorias', 'comboio': 'Conjunto de
    vagões ou carruagens engatadas umas nas outras e puxadas por uma locomotiva', 'mota': 'Veículo
    de duas rodas accionado por um motor'}
# Conjunto de vagões ou carruagens engatadas umas nas outras e puxadas por uma locomotiva
```

Listagem 2.3: Criar dicionario vazio e adicionar itens

2.3 Criar Com Diversos Itens: items(), keys() e values()

```
# https://www.codingame.com/playgrounds/52499/programacao-python-intermediario---prof--marco-vaz/
    dicionarios-continuacao
# Dicionários em Python
# items(), keys() e values()
# Os dicionários são iteráveis, isto é, são objetos que podem ter os elementos da sequência
    percorridos um a um. A função items() retorna uma lista de tuplas com os pares ""chave : valor.
    A função keys() retorna uma lista apenas com as chaves do dicionário, enquanto que a função
    values() retorna uma lista apenas com os valores dos itens do dicionário.
#
cesta_frutas = {"maça": 60, "uva": 20, "pera": 35, "abacaxi": 25, "laranja": 55} 55}
#
print(cesta_frutas.items())
# Out: dict_items([('maça', 60), ('uva', 20), ('pera', 35), ('abacaxi', 25), ('laranja', 55)])
#
print(cesta_frutas.keys())
# Out: dict_keys(['maça', 'uva', 'pera', 'abacaxi', 'laranja'])
#
print(cesta_frutas.values())
# Out: dict_values([60, 20, 35, 25, 55])
#
# Também podemos percorrer os elementos da sequência através da estrutura for, iterando através de
    variáveis que recebem a chave e o valor, definidas no corpo do comando.
#
for fruta, qtd in cesta_frutas.items():
  print('nome:', fruta, (12 - len(fruta)) * ' ', 'qtde:', str(qtd))
# nome: maça          qtde: 60
# nome: uva           qtde: 20
# nome: pera          qtde: 35
# nome: abacaxi       qtde: 25
# nome: laranja       qtde: 55
```

Listagem 2.4: Criar com diversos itens

2.3.1 Dicionários Remover e Atualizar

```
# https://www.stechies.com/difference-between-del-remove-pop-lists/
# The remove function takes an element as an argument and removes
# it from a defined list. If the element 'doesnt exist in the list,
# python throws valueError exception.

# List_name.remove(element)
petlist = ['dog', 'cat', 'mouse', 'rabbit']
petlist.remove('mouse')
print ('Updated list of pets:', petlist)


# List_name.pop(index_no)
#list of Languages
language = ['Hindi', 'English', 'Marathi', 'Bengali', 'urdu']
#Return value from pop()
#When 4 is passed
return_value = language.pop(4)
print('Return Value: ', return_value)
# Updated List
print('Updated List: ', language)

# del List_name(index)

numberslist = [1, 2, 3, 4, 5, 6]
# deleting the third item
del numberslist[2]
print ('the output list :', numberslist )


# Deleting Items from 2nd to 5th
numberslist = [10, 21, 43, 54, 51, 36]
# deleting the third item/slice
del numberslist[2:5]
print ('the output list :', numberslist )



# https://www.educative.io/answers/how-to-remove-a-key-value-pair-from-a-dictionary-in-python
# declare a dictionary
dict1 = {"1": "one", "2": "two", "3": "three", "4": "four"}
print("Original Dictionary: ", dict1)

# using pop()
valDel = dict1.pop("1")
print("Dictionary after using pop(): ", dict1)
print("The value that was removed is: ", valDel)

# using pop()
valDel = dict1.pop("3")
print("Dictionary after using pop(): ", dict1)
print("The value that was removed is: ", valDel)

# using pop()
valDel = dict1.pop("3", "No Key Found")
print("Dictionary after using pop(): ", dict1)
print("The value that was removed is: ", valDel)

# using pop()
# if default value not given, error will be raised
print("Error raised: ")
valDel = dict1.pop("3")
```

Listagem 2.5: listdictremoveupdate.

2.3.2 Qual é a Capital de País Usando Listas

Qual é a capital de Lisboa?

```
# listas: desvantagem
lista_paises_capitais = [('Alemanha', 'Berlim'), ('Áustria', 'Viena'), ('Bélgica', 'Bruxelas'),
                         ('Bielorrússia', 'Minsk'), ('Bulgária', 'Sofia'), ('Chipre', 'Nicósia'),
                         ('Croácia', 'Zagreb'), ('Dinamarca', 'Copenhague'), ('Eslováquia', '
    Bratislava'),
                         ('Espanha', 'Madrid'), ('Finlândia', 'Helsinki'), ('França', 'Paris'), ('Gr
    écia', 'Atenas'),
                         ('Hungria', 'Budapeste'), ('Irlanda', 'Dublin'), ('Itália', 'Roma'), ('Letô
    nia', 'Riga'),
                         ('Mônaco', 'Mônaco'), ('Noruega', 'Oslo'), ('Países Baixos', 'Amsterdão'),
                         ('Polônia', 'Varsóvia'), ('Portugal', 'Lisboa'), ('Tchéquia', 'Praga'),
                         ('Romênia', 'Bucareste'), ('Sérvia', 'Belgrado'), ('Suécia', 'Estocolmo'),
    ('Suíça', 'Berna'),
                         ('Ucrânia', 'Kiev')]

# Qual é a capital de Lisboa?
# É necessária uma pesquisa: ciclo for
def CapitalDePais(lista, pais):
    for x in lista:
        if x[0] == pais:
            return x[1]
    return None

print("-" * 20)
pais = 'Portugal'
capital = CapitalDePais(lista_paises_capitais, pais)
print(f"A capital de {pais} é {capital}.")
# A capital de Portugal é Lisboa.


print("-" * 20)
# Criar um dicionário baseado numa lista com duas colunas.
dic_paises_capitais = dict(lista_paises_capitais)
print(dic_paises_capitais)
```

Listagem 2.6: QualEaCapitalDeListas.

2.3.3 Qual é a Capital de País Usando Dicionário

Qual é a capital de Lisboa?

```
#-----------------
# Criar dicionario:
dic_pais_capital = {'Alemanha': 'Berlim', 'Áustria': 'Viena', 'Bélgica': 'Bruxelas', 'Bielorrússia':
     'Minsk',
                    'Bulgária': 'Sofia', 'Chipre': 'Nicósia', 'Croácia': 'Zagreb', 'Dinamarca': '
    Copenhague',
```

```
                    'Eslováquia': 'Bratislava', 'Espanha': 'Madrid', 'Finlândia': 'Helsinki', 'França': 'Paris',
                    'Grécia': 'Atenas', 'Hungria': 'Budapeste', 'Irlanda': 'Dublin', 'Itália': 'Roma',
                    'Letônia': 'Riga', 'Mônaco': 'Mônaco', 'Noruega': 'Oslo', 'Países Baixos': 'Amsterdão',
                    'Polônia': 'Varsóvia', 'Portugal': 'Lisboa', 'Tchéquia': 'Praga', 'Romênia': 'Bucareste',
                    'Sérvia': 'Belgrado', 'Suécia': 'Estocolmo', 'Suíça': 'Berna', 'Ucrânia': 'Kiev'
                    }


pais = 'Portugal'
capital= dic_pais_capital[pais]
print(f"A capital de {pais} é {capital}.")


print("-" * 20)
# Qual é o País cuja capital é Lisboa?
# Criar um dicionário baseado numa lista com duas colunas.
# Trocar as colunas-chave e valor
col1 = [x for x, y in dic_pais_capital]  # # Python - List Comprehension
col2 = [y for x, y in dic_pais_capital]
dic_capitais_paises = dict(zip(col2, col1))
print(dic_capitais_paises)
capital = 'Lisboa'
pais = dic_capitais_paises[capital]
print(f"O País cuja capital é {capital}, é {pais}.")


capital = 'Lisboa'
pais = dic_pais_capital[capital]
print(f"O País cuja capital é {capital}, é {pais}.")
```

Listagem 2.7: QualEaCapitalDeDicionario.

2.3.4 País Cuja Capital é Usando Dicionário Baseado numa Lista

Qual é o País cuja capital é Lisboa?

Criar um dicionário baseado numa lista com duas colunas.

```
lista_paises_capitais = [('Alemanha', 'Berlim'), ('Áustria', 'Viena'), ('Bélgica', 'Bruxelas'),
                         ('Bielorrússia', 'Minsk'), ('Bulgária', 'Sofia'), ('Chipre', 'Nicósia'),
                         ('Croácia', 'Zagreb'), ('Dinamarca', 'Copenhague'), ('Eslováquia', 'Bratislava'),
                         ('Espanha', 'Madrid'), ('Finlândia', 'Helsinki'), ('França', 'Paris'), ('Grécia', 'Atenas'),
                         ('Hungria', 'Budapeste'), ('Irlanda', 'Dublin'), ('Itália', 'Roma'), ('Letônia', 'Riga'),
                         ('Mônaco', 'Mônaco'), ('Noruega', 'Oslo'), ('Países Baixos', 'Amsterdão'),
                         ('Polônia', 'Varsóvia'), ('Portugal', 'Lisboa'), ('Tchéquia', 'Praga'),
                         ('Romênia', 'Bucareste'), ('Sérvia', 'Belgrado'), ('Suécia', 'Estocolmo'),
                         ('Suíça', 'Berna'),
                         ('Ucrânia', 'Kiev')]

print("-" * 20)
# Qual é o País cuja capital é Lisboa?
# Criar um dicionário baseado numa lista com duas colunas.
```

```
14 # Trocar as colunas-chave e valor
15 col1 = [x for x, y in lista_paises_capitais]  # # Python - List Comprehension
16 col2 = [y for x, y in lista_paises_capitais]
17 dic_capitais_paises = dict(zip(col2, col1))
18 print(dic_capitais_paises)
19 capital = 'Lisboa'
20 pais = dic_capitais_paises[capital]
21 print(f"O País cuja capital é {capital}, é {pais}.")
22 # O País cuja capital é Lisboa, é Portugal.
```

Listagem 2.8: PaisCujaCapitalEDicionario.

2.4 Sets - Conjuntos

https://www.programiz.com/python-programming/set

Um conjunto é uma coleção de dados exclusivos. Ou seja, os elementos de um conjunto não podem ser duplicados. Por exemplo

Suponha que queiramos armazenar informações sobre estudantes. Como os número de estudante não podem ser duplicadas, podemos usar um conjunto.

2.4.1 Sets Exemplos

```
# create a set of integer type
student_id = {112, 114, 116, 118, 115}
print('Student ID:', student_id)

# create a set of string type
vowel_letters = {'a', 'e', 'i', 'o', 'u'}
print('Vowel Letters:', vowel_letters)

# create a set of mixed data types
mixed_set = {'Hello', 101, -2, 'Bye'}
print('Set of mixed data types:', mixed_set)

# Student ID: {112, 114, 115, 116, 118}
# Vowel Letters: {'e', 'i', 'u', 'o', 'a'}
# Set of mixed data types: {'Hello', 'Bye', 101, -2}
```

Listagem 2.9: SetsExemplos.

2.4.2 Criar Sets

```
# create an empty set
empty_set = set()

# create an empty dictionary
empty_dictionary = { }
```

```
# check data type of empty_set
print('Data type of empty_set:', type(empty_set))

# check data type of dictionary_set
print('Data type of empty_dictionary', type(empty_dictionary))
#Data type of empty_set: <class 'set'>
# Data type of empty_dictionary <class 'dict'>
```

Listagem 2.10: CriarSets.

Capítulo 3

Mini-projeto para 2024-01-23e25e30-SA

O objetivo deste projeto é desenvolver um programa que efetue um dado número de questões de escolha múltipla ao utilizador. Cada uma das questões com quatro opções de respostas (A, B, C e D), sendo apenas uma delas correta. O utilizador deve responder a cada uma das questões dentro de um determinado tempo limite. Caso não responda nesse limite, a resposta é considerada errada. O programa deve apresentar estatísticas com o número e percentagem de respostas certas e erradas. Deve também fornecer estatísticas sobre o tempo mínimo, máximo e médio das respostas, das respostas corretas e das respostas erradas.

Todas as questões efetuadas e respostas escolhidas devem ficar armazenadas num ficheiro CSV com o nome `<DataHora _execução>_<nome do utilizador>.CSV`. Por isso, o programa também deve perguntar o nome do utilizador (questionado).

A listagem 3.9 apresenta a saída de ecrã para um exemplo em o tema das questões é sempre sobre capitais de países. Neste caso, o utilizador Ana Esperta respondeu a 6 questões e com um tempo limite de 10 segundos.

```
Teste de Cultura Sobre Capitais de Países

Indique o seu nome? Ana Esperta
Quantas questões pretende responder? 6
================
= Questão 1    =
******************************
Qual é a capital de Eslováquia?
A - Bratislava.
B - Helsinki.
C - Paris.
D - Atenas.
******************************
Hora: 20:59:14
Atenção: Tem 10 segundos para responder à questão:
Resposta (A,B,C,D)? a
Tempo:  1.6
================
```

```
= Questão 2    =
*****************************
Qual é a capital de Portugal?
A - Lisboa.
B - Berna.
C - Kiev.
D - Berlim.
*****************************
Hora: 20:59:15
Atenção: Tem 10 segundos para responder à questão:
Resposta (A,B,C,D)? a
Tempo:  1.51
================
= Questão 3    =
*****************************
Qual é a capital de Suécia?
A - Estocolmo.
B - Berna.
C - Kiev.
D - Berlim.
*****************************
Hora: 20:59:17
Atenção: Tem 10 segundos para responder à questão:
Resposta (A,B,C,D)? a
Tempo:  3.93
================
= Questão 4    =
*****************************
Qual é a capital de Polônia?
A - Varsóvia.
B - Lisboa.
C - Praga.
D - Bucareste.
*****************************
Hora: 20:59:21
Atenção: Tem 10 segundos para responder à questão:
Resposta (A,B,C,D)? a
Tempo:  2.08
================
= Questão 5    =
*****************************
Qual é a capital de Portugal?
A - Lisboa.
B - Praga.
C - Bucareste.
D - Belgrado.
*****************************
Hora: 20:59:23
Atenção: Tem 10 segundos para responder à questão:
Resposta (A,B,C,D)? a
Tempo:  1.54
================
= Questão 6    =
*****************************
Qual é a capital de Ucrânia?
A - Kiev.
B - Berlim.
C - Viena.
D - Bruxelas.
*****************************
Hora: 20:59:24
Atenção: Tem 10 segundos para responder à questão:
Resposta (A,B,C,D)? a
Tempo:  14.03
```

```
Excedeu o tempo limite de 10 segundos.
Por isso, a sua respostas é considerada errada.


Resultados
(1, 'Qual é a capital de Eslováquia? ', [('Bratislava', 'A', 1), ('Helsinki', 'B', 0), ('Paris', 'C'
    , 0), ('Atenas', 'D', 0)], 'A', 1.6, True, 'Correta dentro do tempo de resposta')
(2, 'Qual é a capital de Portugal? ', [('Lisboa', 'A', 1), ('Berna', 'B', 0), ('Kiev', 'C', 0), ('
    Berlim', 'D', 0)], 'A', 1.51, True, 'Correta dentro do tempo de resposta')
(3, 'Qual é a capital de Suécia? ', [('Estocolmo', 'A', 1), ('Berna', 'B', 0), ('Kiev', 'C', 0), ('
    Berlim', 'D', 0)], 'A', 3.93, True, 'Correta dentro do tempo de resposta')
(4, 'Qual é a capital de Polônia? ', [('Varsóvia', 'A', 1), ('Lisboa', 'B', 0), ('Praga', 'C', 0), (
    'Bucareste', 'D', 0)], 'A', 2.08, True, 'Correta dentro do tempo de resposta')
(5, 'Qual é a capital de Portugal? ', [('Lisboa', 'A', 1), ('Praga', 'B', 0), ('Bucareste', 'C', 0),
     ('Belgrado', 'D', 0)], 'A', 1.54, True, 'Correta dentro do tempo de resposta')
(6, 'Qual é a capital de Ucrânia? ', [('Kiev', 'A', 1), ('Berlim', 'B', 0), ('Viena', 'C', 0), ('
    Bruxelas', 'D', 0)], 'A', 14.03, False, 'Excedeu tempo de resposta.')
{'Corretas': 5, 'Erradas': 1, 'Percentagem Corretas': 83.3, 'Percentagem Erradas':
    16.700000000000003}
```

Listagem 3.1: Exemplo de interação do utilizador com o programa

3.1 Parte I

Desenvolver um prototipo para simular o caso da listagem 3.9.

Definir estruturas de dados:

1. Questões e respostas.
2. Respostas do utilizador às questões.
3. Resultados.
4. Estatísticas.

Organizar o código:

1. Programa principal
2. Diversos módulos com funções
3.

Repare que:

1. O número de carateres * varia de acordo com a pergunta.
2. ...

Melhoramentos:

1. Determinar se o nome do país é masculino ou feminino. Fazer a pergunta de acordo com o resultado.
2. Colocar duas (ou três, etc) questões por linha em vez de estarem as quatro.
3. Baralhar as respostas. Ou seja, as opções devem estar em posições diferentes cada vez que o programa é executado.

3.2 Parte II

Desenvolver uma versão II do prototipo I em que as questões e as respostas são obtidas do ficheiros CSV `paises_geonames.com.txt`.

No ficheiro existem 252 países. De acordo com a especificação do programa, responder a N questões, deve escolher apenas N países do ficheiro. Depois, formula as questões acerca de diversos assuntos, que podem também ser selecionados aleatoriamente. l

A Listagem 3.2 lista um programa para simular o lançamento de um dado de seis faces. Pode usar o mesmo processo para escolher N países da lista de todos os países.

```
import random

while True:
    op = input("Enter para lançar dado (x - terminar): ")
    if op == 'x':
        break
    n = random.randint(1, 6)
    print("Face:", n)

# Enter para lançar dado (x - terminar):
# Face: 6
# Enter para lançar dado (x - terminar):
# Face: 1
# Enter para lançar dado (x - terminar):
# Face: 4
# Enter para lançar dado (x - terminar):
# Face: 5
# Enter para lançar dado (x - terminar):
# Face: 5
# Enter para lançar dado (x - terminar):
# Face: 2
# Enter para lançar dado (x - terminar):
```

Listagem 3.2: AleatorioDadosSeisFaces.

Pode criar perguntas sobre diversos assuntos de acordo com as colunas de dados existentes no ficheiros:

```
# 0 - ISO
# 1 - ISO3
# 2 - ISO-Numeric
# 3 - fips
# 4 - Country
# 5 - Capital
```

```
# 6 - Area(in sq km)
# 7 - Population
# 8 - Continent
# 9 - tld
# 10 - CurrencyCode
# 11 - CurrencyName
# 12 - Phone
# 13 - Postal Code Format
# 14 - Postal Code Regex
# 15 - Languages
# 16 - geonameid
# 17 - neighbours
# 18 - EquivalentFipsCode
```

Listagem 3.3: Colunas no ficheiro `paises_geonames.com.txt`.

Por exemplo pode perguntar acerca da população dos países:

```
Qual dos países tem cerca de 10 milhões de habitantes?
 A - Portugal
 B - Espanha
 C - França
 D - Itália
```

Listagem 3.4: questão sobre população em países.

Outro tipo de pergunta e acerca dos vizinho dos países. Por exemplo,

```
Qual dos países não é vizinho de Espanha?
  A - Andorra
  B - France
  C - Portugal
  D - Itália
```

Listagem 3.5: questão sobre população em países.

3.2.1 LerEscreverCSVDict

```
1  import csv
2  def LerCSVParaListadeDicionarios(nome_ficheiro, delimitador=';'):
3      dados = {}
4      with open(nome_ficheiro, 'r', newline='', encoding='utf-8') as f:
5          reader = csv.DictReader(f, delimiter=delimitador)
6          # for r in reader:
7          #     lista[r['']] = r  # load all records - dict of dicts
8          dic = [r for r in reader] # load all records - list of dicts
9      return dic
10
11 def LerCSVParaDicionariodeDicionarios(nome_ficheiro, chave, delimitador=';'):
12     dados = {}
13     with open(nome_ficheiro, 'r', newline='', encoding='utf-8') as f:
14         reader = csv.DictReader(f, delimiter=delimitador)
15         dic_dic = {}
16         for r in reader:
17              dic_dic[r[chave]] = r  # load all records - dict of dicts
18         # lista = [r for r in reader]  # load all records - list of dicts
19     return dic_dic
20
```

```python
def GravarListaDeDicionarioCSV(lista_dicionarios, nome_ficheiro, delimitador=';'):
    with open(nome_ficheiro, 'w', newline='', encoding='utf-8') as csvfile:
        # a) 1 linha da lista. b) chaves da linha. c) conversão para lista
        titulos = list(lista_dicionarios[0].keys())
        writer = csv.DictWriter(csvfile, delimiter=delimitador, fieldnames=titulos)
        writer.writeheader()
        writer.writerows(lista_dicionarios)
        # for r in lista_dicionarios:
        #     writer.writerow(r)

def GravarDicionarioDeDicionarioCSV(dicionario_dicionarios, nome_ficheiro, delimitador=';'):
    with open(nome_ficheiro, 'w', newline='', encoding='utf-8') as csvfile:
        # a) chaves para lista e chave1. b) 1o dicionario da chave1.
        # c) chaves da chave1 e conversão para lista
        chave1 = list(dicionario_dicionarios.keys())[0]
        dic1 = dicionario_dicionarios[chave1]
        titulos = list(dic1.keys())
        writer = csv.DictWriter(csvfile, delimiter=delimitador, fieldnames=titulos)
        writer.writeheader()
        for d in dicionario_dicionarios.values():
             writer.writerow(d)
```

Listagem 3.6: LerEscreverCSVDict.

3.2.2 LerGravarCSV

```python
def GravarListaEmFicheiroCSV(lista, nome_ficheiro, delimitador=';', titulos=None):
    """
    :param lista:
    :param nome_ficheiro:
    :param delimitador:
    :param titulos:
    :return:
    """
    import csv
    with open(nome_ficheiro, 'w', newline='', encoding='utf-8') as f:
        writer = csv.writer(f, delimiter=delimitador)
        if titulos is not None:
            writer.writerow(titulos)
        writer.writerows(lista)


def LerFicheiroCSVParaLista(nome_ficheiro, delimitador,tem_titulos=False):
    """
    :param nome_ficheiro:
    :param delimitador:
    :param tem_titulos:
    :return:
    """
    import csv
    lista = []
    titulos = []
    with open(nome_ficheiro, 'rt', newline='', encoding='utf-8') as f:
        reader = csv.reader(f, delimiter=delimitador)
        if tem_titulos:
            # Skips the heading:    # Using next() method
            titulos = next(reader)  # ou titulos = reader.__next__()
        for r in reader:
            lista.append(r)
    return titulos, lista
```

Listagem 3.7: LerGravarCSV.

3.2.3 Ler Ficheiro CSV para Lista e Dicionário

```
from LerEscreverCSVDict import *

lista_de_dicionarios = LerCSVParaListadeDicionarios('paises_geonames.com.txt', delimitador='\t')
# print(dicionario)
print('LerCSVParaListadeDicionarios:', "*" * 50)
for x in lista_de_dicionarios:
print(x)
break
GravarListaDeDicionarioCSV(lista_de_dicionarios, "gravar_lista_dics.csv", delimitador=';')
# teste que gravou bem.
# lista_de_dicionarios = LerCSVParaListadeDicionarios('gravar_lista_dics.csv', delimitador=';')
# print(lista_de_dicionarios)

# ----------
dicionario_de_dicionarios = LerCSVParaDicionariodeDicionarios('paises_geonames.com.txt', 'Country',
    delimitador='\t')
print('LerCSVParaDicionariodeDicionarios:', "*" * 50)
for chave, valor in dicionario_de_dicionarios.items():
print(f"{chave}: {valor}")
break
GravarDicionarioDeDicionarioCSV(dicionario_de_dicionarios, "gravar_dic_dics.csv", delimitador=';')
# teste que gravou bem.
# dicionario_de_dicionarios = LerCSVParaDicionariodeDicionarios('gravar_dic_dics.csv','Country',
    delimitador=';')
# print(dicionario_de_dicionarios)
```

Listagem 3.8: Exemplo para ler ficheiro o `paises_geonames.com.txt` para lista e dicionário.

3.3 Parte III

Adicionar voz ao prototipo II para efetuar as questões e respostas.

```
def Coffee():
  import pyttsx3
  engine = pyttsx3.init()
  import Cores
  import time

  pergunta = "Coffee break?"
  print(pergunta, end='')
  engine.say(pergunta)
  engine.runAndWait()

  pergunta = "Yes. At 10:30 am"
  print(pergunta, end='')
  engine.say(pergunta)
  engine.runAndWait()
Coffee()
```

Listagem 3.9: Exemplo de interação do utilizador com o programa

Bibliografia

[1] python.org. dictionaries. https://docs.python.org/3/tutorial/datastructures.html#dictionaries, Jan 2023.