Curso:

**UPskill – Programação em Python**

Unidade de Formação:

**Programação**

Trabalhos SA

**Ementas\_Avaliações**

Henrique José Alves Gonzaga

Entrega:

**2024-01-19 23:59**

Formato de entrega:

**PDF, baseado no formato existente no moodle (formato trabalho 1)**

Índice

[1 Introdução 1](#_Toc156552426)

[2 Programa 2](#_Toc156552427)

[2.1 Ementa 2](#_Toc156552428)

[2.2 Atendimentos 2](#_Toc156552429)

[3 Casos de Teste 5](#_Toc156552430)

[3.1 Caso 1 5](#_Toc156552431)

[3.1.1 Ementas 5](#_Toc156552432)

[3.1.2 Atendimentos 5](#_Toc156552433)

[3.2 Caso 2 5](#_Toc156552434)

[3.2.1 Ementas 5](#_Toc156552435)

[3.2.2 Atendimento 6](#_Toc156552436)

[3.3 Caso 3 6](#_Toc156552437)

[3.3.1 Ementas 6](#_Toc156552438)

[3.4 Caso 4 6](#_Toc156552439)

[3.4.1 Ementas 6](#_Toc156552440)

[**3.5** **Caso 5** 7](#_Toc156552441)

[3.5.1 Ementas 7](#_Toc156552442)

[**3.6** **Caso 6** 7](#_Toc156552443)

[3.6.1 Ementas 7](#_Toc156552444)

[**3.7** **Caso 7** 7](#_Toc156552445)

[3.7.1 Ementas 7](#_Toc156552446)

[**3.8** **Caso 8** 7](#_Toc156552447)

[3.8.1 Ementas 7](#_Toc156552448)

[**3.9** **Caso 9** 7](#_Toc156552449)

[3.9.1 Ementas 7](#_Toc156552450)

[**3.10** **Caso 10** 7](#_Toc156552451)

[3.10.1 Ementas 7](#_Toc156552452)

# Introdução

Foram feitos dois programas, um para receber uma data e retornar a semana em que a ementa será feita. No segundo, é recebido uma lista com os dados dos exames, exames recurso e frequência e a partir dai calcular os dias dos atendimentos evitando as colisões.

# Programa

## Ementa

from datetime import datetime, timedelta  
import locale  
  
locale.setlocale(locale.LC\_ALL, '')  
data\_input = input("informe uma data (dd/mm/aaaa): ")  
data\_formatada = datetime.strptime(data\_input, '%d/%m/%Y')  
data\_seg = data\_formatada  
  
while True:  
 dia\_semana = int(datetime.strftime(data\_seg, '%w')) + 1  
 if dia\_semana == 2:  
 break  
 else:  
 data\_seg = data\_seg - timedelta(days=1)  
data\_sex = data\_seg + timedelta(days=4)  
  
print(datetime.strftime(data\_seg, 'Semana de %d a'), end=' ')  
print(datetime.strftime(data\_sex, '%d de %B de %Y'))  
locale.setlocale(locale.LC\_ALL, 'en\_US')  
print(datetime.strftime(data\_seg, 'Week from %d to'), end=' ')  
print(datetime.strftime(data\_sex, '%dth %B, %Y'))

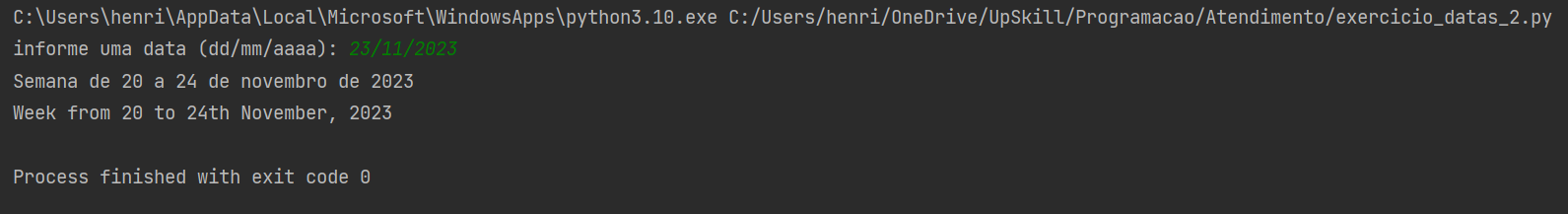
## Atendimentos

from datetime import datetime, timedelta  
import locale  
  
locale.setlocale(locale.LC\_ALL, '')  
cursos = {'TC': 'TESP CiberSegurança', 'LEI': 'Licenciatura em Engenharia Informática '}  
disciplinas = {'PI': 'Programação para a Internet ', 'AED': 'Algoritmos e Estruturas de Dados '}  
tipos = {'F': '\* Frequência', 'E': '\* Exame ', "R": '\* Recurso ', 'ATM': 'Atendimento '}  
resultado = {'curso': '', 'disciplina': '', 'data': '', 'hora\_inicio': '', 'hora\_fim': '', 'tipologia': ''}  
  
datasTexto = [  
 ('LCDIA', 'AA1', '02/02/2024 14:00', 'E'),  
 ('LCDIA', 'AA1', '08/02/2024 10:00', 'R'),  
 ('LEI', 'PI', '05/02/2024 17:00', 'E'),  
 ('LEI', 'PI', '10/02/2024 10:00', 'R')  
]  
  
datasTexto1 = [  
 ('LEI', 'EE', '17/01/2024 14:00', 'F'),  
 ('LEI', 'EE', '03/02/2024 14:00', 'E'),  
 ('LEI', 'EE', '14/02/2024 17:00', 'R'),  
 ('MCM', 'RSC', '24/01/2024 14:00', 'F'),  
 ('MCM', 'RSC', '02/02/2024 14:00', 'E'),  
 ('MCM', 'RSC', '10/02/2024 11:00', 'R'),  
 ('TESPC', 'RC2', '17/01/2024 14:00', 'E'),  
 ('TESPC', 'RC2', '03/02/2024 14:00', 'F'),  
 ('TESPC', 'RC2', '14/02/2024 14:00', 'R')  
]  
  
  
def data\_dia\_anterior(data, dias):  
 data\_anterior = data  
 data\_anterior -= timedelta(days=dias)  
 return data\_anterior  
  
  
def verifica\_data(data\_anterior):  
 verifica\_data\_anterior = int(datetime.strftime(data\_anterior, '%w'))  
 if verifica\_data\_anterior == 6:  
 data\_anterior -= timedelta(days=2)  
 elif verifica\_data\_anterior == 0:  
 data\_anterior -= timedelta(days=2)  
 return data\_anterior  
  
  
def verifica\_dia\_numeral\_semana(dia):  
 verifica\_semana = int(datetime.strftime(dia, '%w'))  
 return verifica\_semana  
  
  
def horario\_atendimento(hora):  
 if hora <= 12:  
 hora\_inicio = hora\_exame - timedelta(hours=1)  
 else:  
 hora\_inicio = hora\_exame  
 hora\_final = hora\_inicio + (timedelta(hours=3))  
 return hora\_final  
  
  
db = datasTexto1  
datas\_atm = []  
for i in db:  
 tipo = i[3]  
 get\_data = i[2].split(' ')  
 get\_hora\_exame = get\_data[1].split(':')  
 hora = int(get\_hora\_exame[0])  
 hora\_exame = timedelta(hours=int(get\_hora\_exame[0]), minutes=int(get\_hora\_exame[1]))  
 if hora <= 12:  
 hora\_inicio\_atendimento = hora\_exame - timedelta(hours=1)  
 else:  
 hora\_inicio\_atendimento = hora\_exame  
 hora\_final\_atendimento = hora\_inicio\_atendimento + (timedelta(hours=3))  
 hora\_final\_exame = hora\_exame + timedelta(hours=2)  
  
 data\_exame = datetime.strptime(get\_data[0], '%d/%m/%Y')  
 verifica\_data\_exame = verifica\_data(data\_exame)  
 verifica\_dia\_semana\_exame = verifica\_dia\_numeral\_semana(data\_exame)  
  
 data\_anterior2 = data\_dia\_anterior(data\_exame, 2)  
 verifica\_data\_anterior2 = verifica\_data(data\_anterior2)  
 verifica\_dia\_semana\_atendimento2 = verifica\_dia\_numeral\_semana(data\_anterior2)  
  
 data\_anterior1 = data\_dia\_anterior(data\_exame, 1)  
 verifica\_data\_anterior1 = verifica\_data(data\_anterior1)  
 verifica\_dia\_semana\_atendimento1 = verifica\_dia\_numeral\_semana(data\_anterior1)  
  
 if verifica\_data\_exame == verifica\_data\_anterior2:  
 data\_exame += timedelta(days=2)  
 verifica\_data\_exame = verifica\_data(data\_exame)  
 if hora\_exame <= timedelta(hours=12):  
  
 hora\_inicio\_atendimento = timedelta(hours=14, minutes=0)  
 hora\_final\_atendimento = hora\_inicio\_atendimento + (timedelta(hours=3))  
 else:  
 hora\_inicio\_atendimento = timedelta(hours=9, minutes=0)  
 hora\_final\_atendimento = hora\_inicio\_atendimento + (timedelta(hours=3))  
  
 elif verifica\_data\_exame == verifica\_data\_anterior1:  
 data\_exame += timedelta(days=1)  
 verifica\_data\_exame = verifica\_data(data\_exame)  
 if hora\_exame <= timedelta(hours=12):  
 hora\_inicio\_atendimento = timedelta(hours=14, minutes=0)  
 hora\_final\_atendimento = hora\_inicio\_atendimento + (timedelta(hours=3))  
 else:  
 hora\_inicio\_atendimento = timedelta(hours=9, minutes=0)  
 hora\_final\_atendimento = hora\_inicio\_atendimento + (timedelta(hours=3))  
  
 data\_exame\_formatada = datetime.strftime(verifica\_data\_exame, '%a %d/%m/%Y')  
  
 data\_anterior2\_formatada = datetime.strftime(verifica\_data\_anterior2, '%a %d/%m/%Y')  
  
 data\_anterior1\_formatada = datetime.strftime(verifica\_data\_anterior1, '%a %d/%m/%Y')  
  
 datas\_atm.append([  
 i[0], i[1], data\_anterior2\_formatada, str(hora\_inicio\_atendimento), str(hora\_final\_atendimento), tipos['ATM']  
 ])  
 datas\_atm.append([  
 i[0], i[1], data\_anterior1\_formatada, str(hora\_inicio\_atendimento), str(hora\_final\_atendimento), tipos['ATM']  
 ])  
 datas\_atm.append([i[0], i[1], data\_exame\_formatada, str(hora\_exame), str(hora\_final\_exame), i[3]])  
  
L = 10  
  
lista\_ordenada = sorted(datas\_atm, key=lambda x: x[2])  
  
for k in resultado.keys():  
 print(k.ljust(L), end=' ' \* L)  
print()  
  
for i, v in enumerate(lista\_ordenada):  
 for j in v:  
 print(j.ljust(L), end=' ' \* L)  
 print()

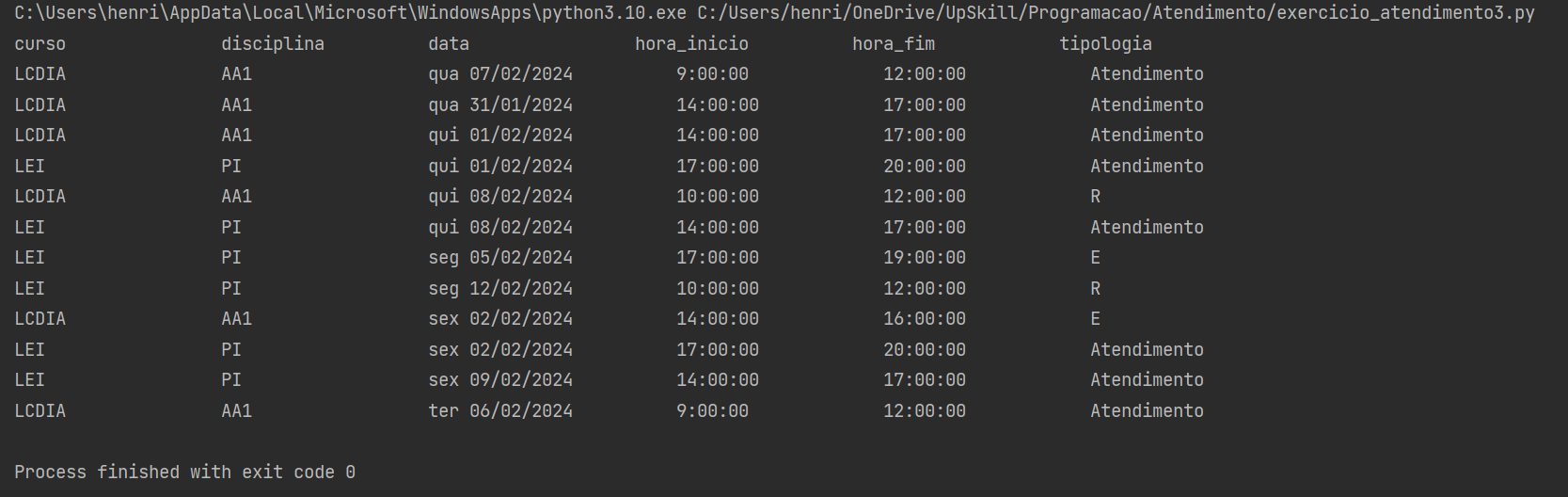
# Casos de Teste

## Caso 1

### Ementas

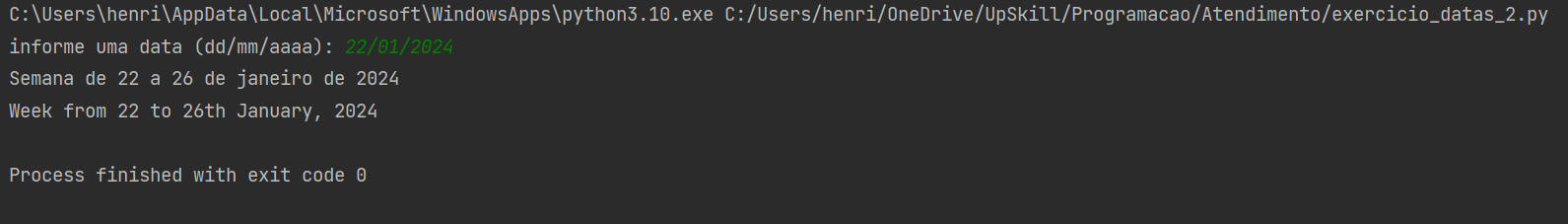


### Atendimentos

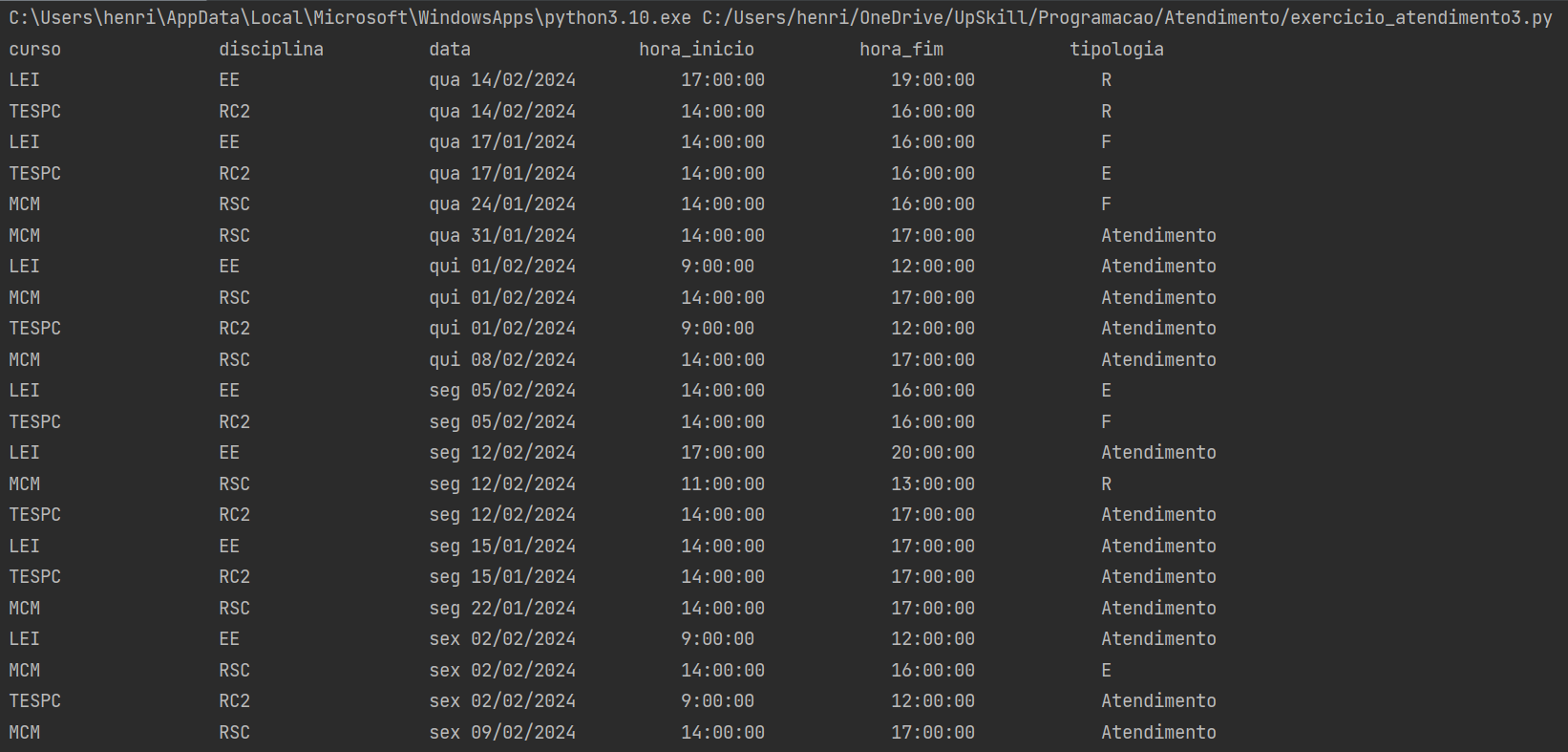


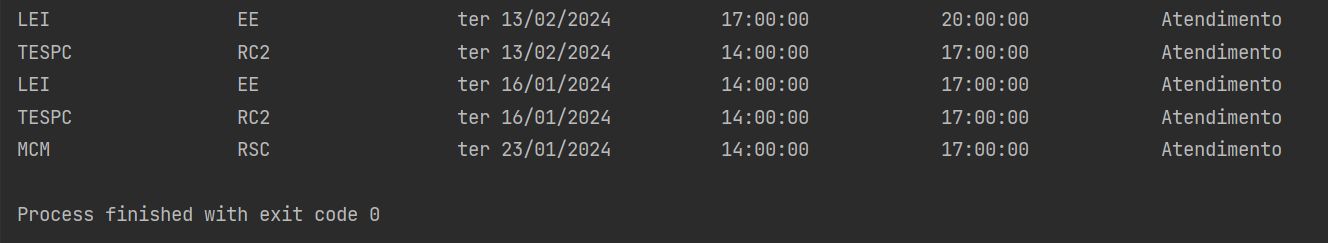
## Caso 2

### Ementas



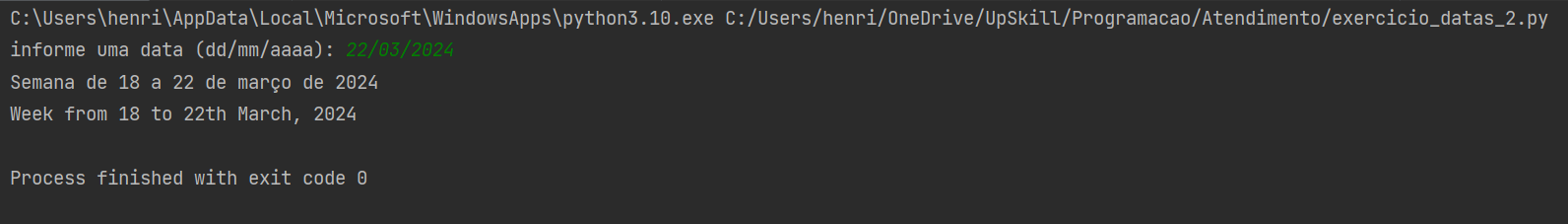
### Atendimento





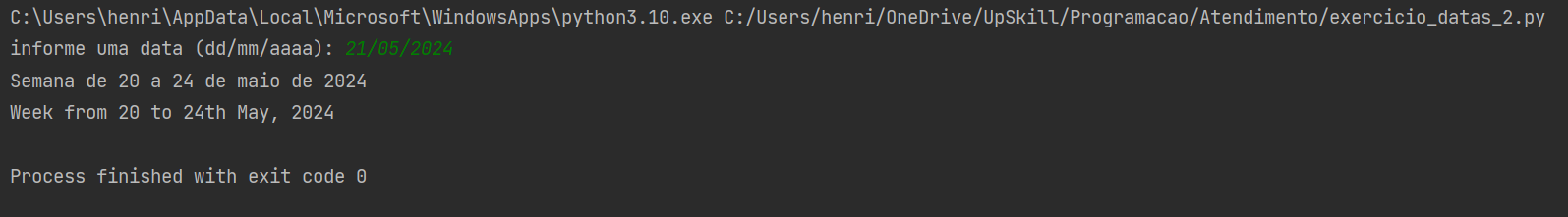
## Caso 3

### Ementas



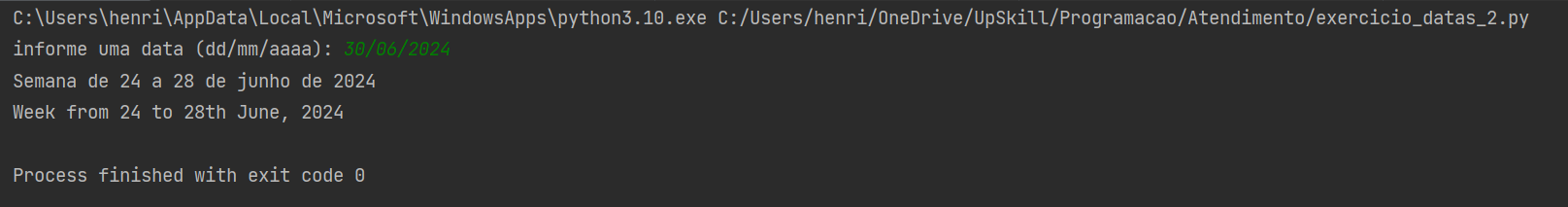
## Caso 4

### Ementas



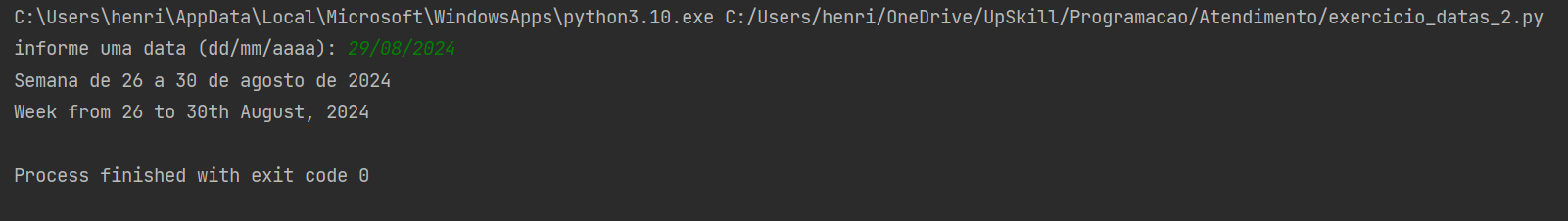
## **Caso 5**

### Ementas



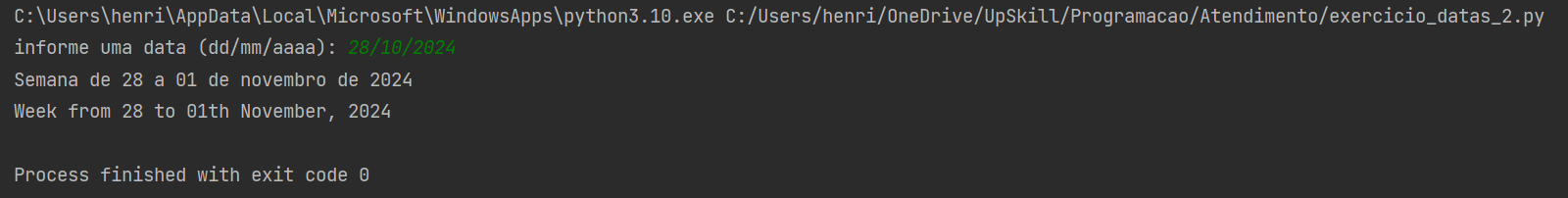
## **Caso 6**

### Ementas



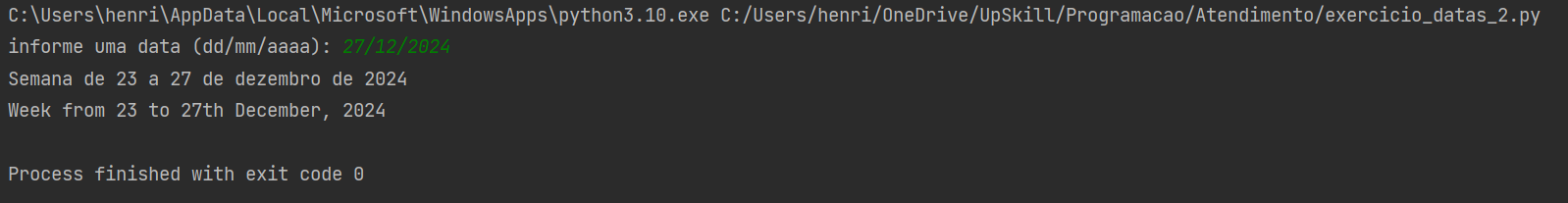
## **Caso 7**

### Ementas



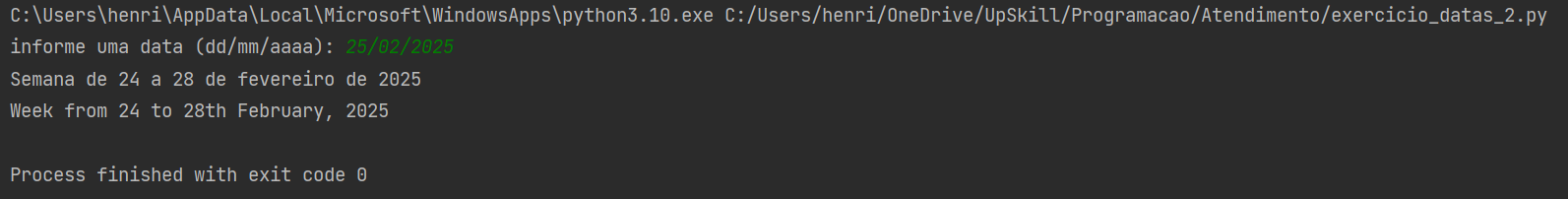
## **Caso 8**

### Ementas



## **Caso 9**

### Ementas



## **Caso 10**

### Ementas

