Instituto INFNET

ESTI - Escola Superior da Tecnologia da Informação

Discente: William Felicio Freire
Docente: Thaís Viana
Disciplina: Fundamentos do Desenvolvimento Python
Assessment

08/12/2020

1. Usando o Thonny, escreva um programa em Python que leia uma tupla contendo 3 números inteiros, (n1, n2, n3) e os imprima em ordem crescente.

```
n1 = int(input("insira um numero inteiro: "))
n2 = int(input("insira um numero inteiro: "))
n3 = int(input("insira um numero inteiro: "))

tuple = (n1, n2, n3)
tuple_ordenada = sorted(tuple)

print(tuple_ordenada)
```

```
PS D:\dev\python\at-py> python3 .\tupla1.py
insira um numero inteiro: 3
insira um numero inteiro: 4
insira um numero inteiro: 5
[3, 4, 5]
```

2. Usando o Thonny, escreva um programa em Python que some todos os números pares de 1 até um dado n, inclusive. O dado n deve ser obtido do usuário. No final, escreva o valor do resultado desta soma.

```
n = int(input("insira um numero inteiro: "))
lista = []
result = 0
for i in range(1, n + 1):
    if i % 2 == 0:
        lista.append(i)
    result = sum(lista)

print(result)
```

```
PS D:\dev\python\at-py> python3 .\sum_pair_range2.py
insira um numero inteiro: 8
20
```

3. Usando o Thonny, escreva uma função em Python chamada potência. Esta função deve obter como argumentos dois números inteiros, A e B, e calcular AB usando multiplicações sucessivas (não use a função de python math.pow) e retornar o resultado da operação. Depois, crie um programa em Python que obtenha dois números inteiros do usuário e indique o resultado de AB usando a função

```python
def potencia(a, b):
    return (a**b)


n1 = int(input("insira um numero inteiro: "))
n2 = int(input("insira um numero inteiro: "))

if n1 and n2 >= 0:
    print('O resultado é', potencia(n1, n2))
else:
    print("Digite numeros inteiros positivos")
```

```
PS D:\dev\python\at-py> python3 .\potencia3.py
insira um numero inteiro: 8
insira um numero inteiro: 4
O resultado é 4096
```

4. Escreva um programa em Python que leia um vetor de 5 números inteiros e o apresente na ordem inversa. Imprima o vetor no final. Use listas.

```python
list = [32, 89, 15, 99, 2]

list.reverse()

print(list)
```

5. ANULADA

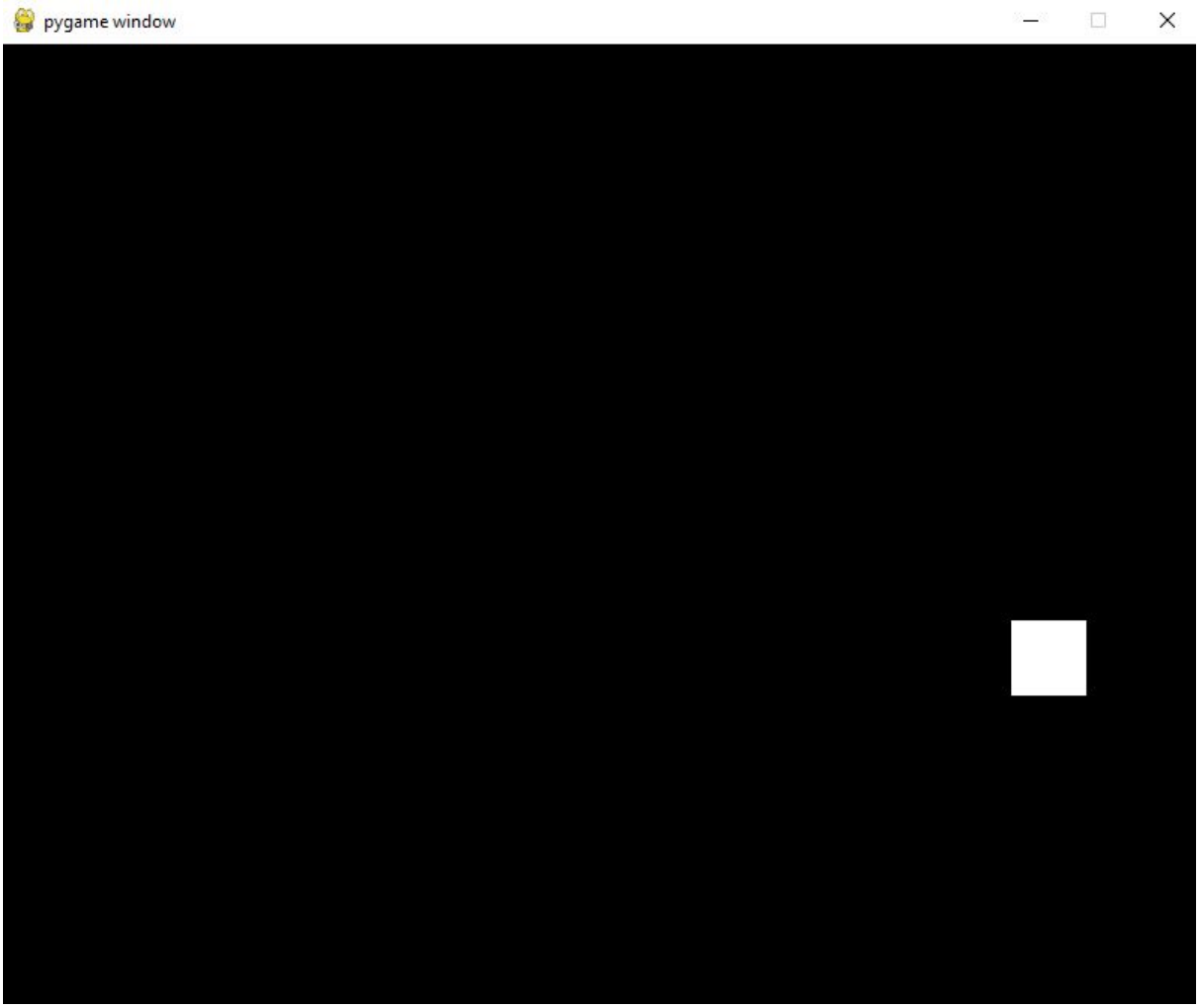
6. Escreva uma função em Python que leia uma tupla contendo números inteiros, retorne uma lista contendo somente os números ímpares e uma nova tupla contendo somente os elementos nas posições pares.

```
tuple_number = (32, 89, 15, 99, 2)
list_pais = []
list_evens = []
for i in tuple_number:
    if i % 2 == 0:
        list_pais.append(i)
    else:
        list_evens.append(i)


print("Números Ímpares: ", list_evens)
print("Números Pares: ", tuple(list_pais))
```

7. Usando a biblioteca ‘pygame’, escreva um programa que desenha na tela em posição aleatória um quadrado amarelo de tamanho 50 (cinquenta), toda vez que a tecla espaço for pressionada ou o botão direito for clicado.

```
import pygame
import random

pygame.init()
largura_tela, altura_tela = 800, 640
tela = pygame.display.set_mode((largura_tela, altura_tela))
clock = pygame.time.Clock()

branco = (255, 255, 255)
preto = (0, 0, 0)
```

```python
def generate_random():
    return [random.randint(0, 750), random.randint(0, 590)]


def draw_quadrado(x, y):
    tela.fill(preto)
    area = pygame.Rect(x, y, 50, 50)
    pygame.draw.rect(tela, branco, area)
    pygame.display.update()


terminou = False
posicao_atual = [largura_tela/2 - 50, altura_tela/2 - 50]
draw_quadrado(*posicao_atual)

while not terminou:
    # Checar os eventos do mouse aqui
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            terminou = True

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_SPACE:
                larg, altu = generate_random()
                draw_quadrado(larg, altu)


# Finaliza a janela do jogo
pygame.display.quit()
# Finaliza o pygame
pygame.quit()
```

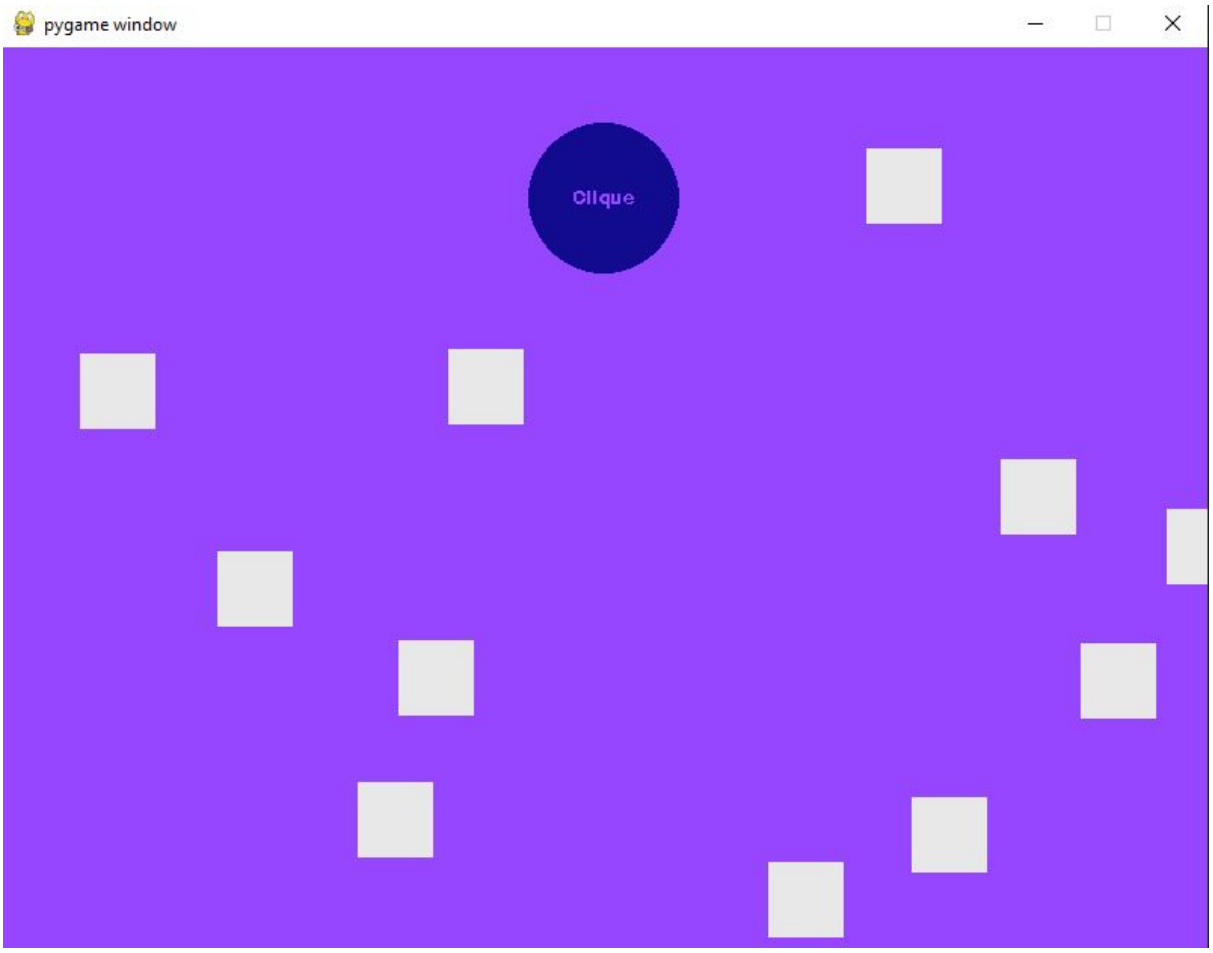


8. Usando a biblioteca ‘pygame’, escreva um programa que desenha um botão (círculo) com o texto “clique” sobre ele na parte superior da tela. Quando o botão for clicado, ele deve chamar uma função que desenha um retângulo em uma posição aleatória na tela. Caso um retângulo apareça na mesma posição que um já existente, ambos devem ser eliminados.

```
import pygame
import sys
import random
from collections import namedtuple


class Collision:
```

```
    Coordinate = namedtuple('Coordinate', ['x', 'y'])
    running = True
    background = (150, 70, 255)
    cinza = (232, 232, 232)
    azul = (18, 10, 143)

    quadradinhos = []

    def __init__(self):
        pygame.init()
        pygame.font.init()

        self.font = pygame.font.Font(None, 20)
        self.screenSize = self.Coordinate(x=800, y=600)
        self.screen = pygame.display.set_mode(self.screenSize)

        self.button = pygame.Rect(
            self.screen.get_width() // 2 - 50, 50, 100, 100)

        self.loop()

    def loop(self):
        while self.running:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    self.running = False
                    sys.exit()
                if event.type == pygame.MOUSEBUTTONDOWN:
                    self.handle_mouse(event)

            self.screen.fill(self.background)

            self.draw_circle('Clique')

            for quadrado in self.quadradinhos:
                pygame.draw.rect(self.screen, self.cinza, quadrado)

            pygame.display.update()

        pygame.display.quit()

    def handle_mouse(self, event):
        if event.button == 1:
            position = pygame.mouse.get_pos()

            if self.button.collidepoint(position):
```

```
                rect = self.draw_square()
                self.quadradinhos.append(rect)
                self.verifica_colisao(rect)

    def draw_circle(self, text):
        pygame.draw.ellipse(self.screen, self.azul, self.button)
        button_text = self.font.render(text, False, self.background)
        button_text_rect =
button_text.get_rect(center=self.button.center)

        self.screen.blit(button_text, button_text_rect)

    def draw_square(self):
        x = random.randint(25, self.screen.get_width() - 25)
        y = random.randint(25, self.screen.get_height() - 25)

        return pygame.Rect(x, y, 50, 50)

    def verifica_colisao(self, rect):
        for quadrado in self.quadradinhos:
            if quadrado is not rect and quadrado.colliderect(rect):
                self.quadradinhos.remove(quadrado)
                if rect in self.quadradinhos:
                    self.quadradinhos.remove(rect)


colision_game = Collision()
```

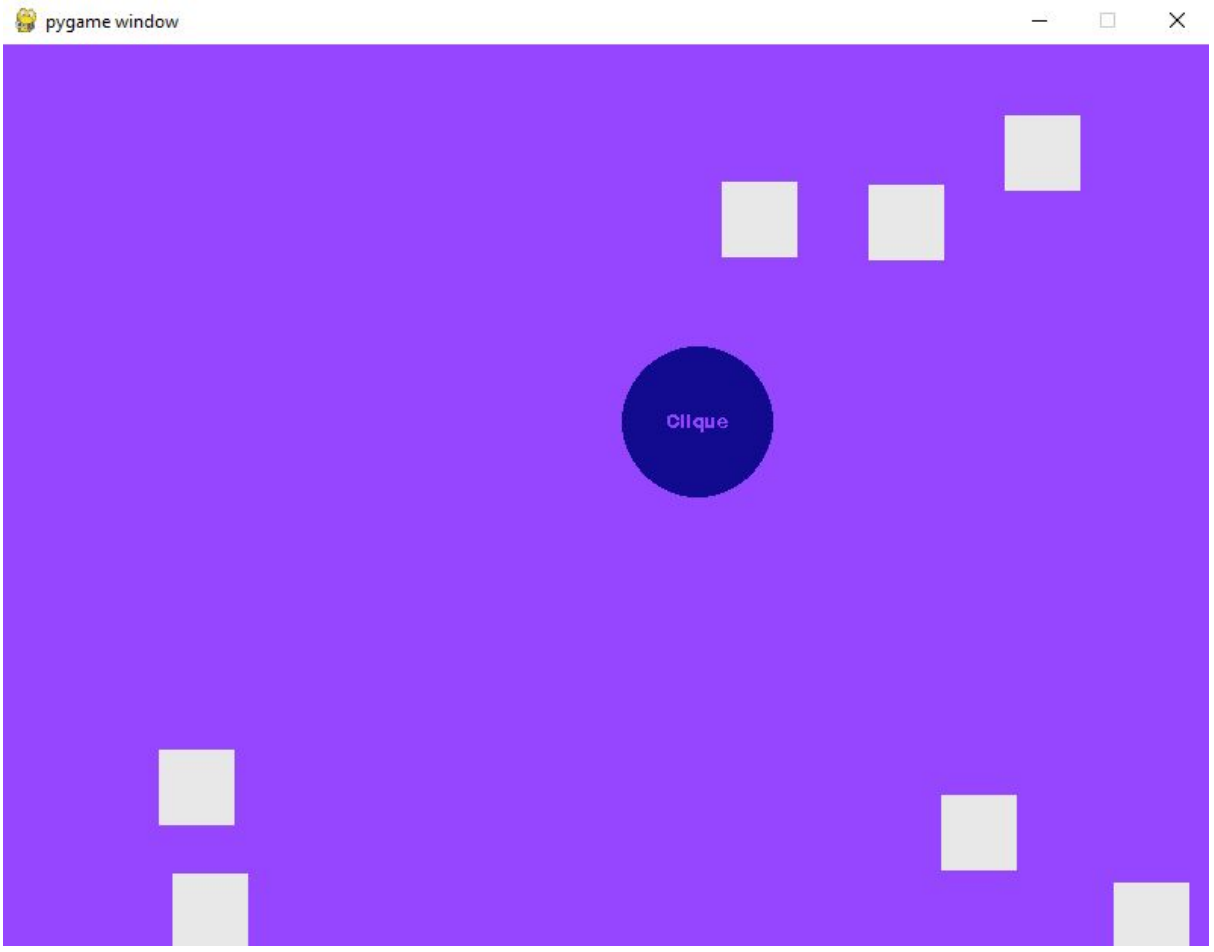


9. Usando o código anterior, escreva um novo programa que, quando as teclas ‘w’, ‘a’, ‘s’ e ‘d’ forem pressionadas, ele movimente o círculo com o texto “clique” nas direções corretas. Caso colida com algum retângulo, o retângulo que participou da colisão deve desaparecer.

```
import pygame
import sys
import random
from collections import namedtuple


class Collision:
    Coordinate = namedtuple('Coordinate', ['x', 'y'])
    running = True
    background = (150, 70, 255)
    cinza = (232, 232, 232)
    azul = (18, 10, 143)
```

```
    speed = 5

    quadradinhos = []

    def __init__(self):
        pygame.init()
        pygame.font.init()

        self.font = pygame.font.Font(None, 20)
        self.screenSize = self.Coordinate(x=800, y=600)
        self.screen = pygame.display.set_mode(self.screenSize)

        self.button = pygame.Rect(
            self.screen.get_width() // 2 - 50, 50, 100, 100)

        self.loop()

    def loop(self):
        while self.running:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    self.running = False
                    sys.exit()
                if event.type == pygame.MOUSEBUTTONDOWN:
                    self.create_quadradinho(event)
                if event.type == pygame.KEYDOWN:
                    self.walk_circle(event)

            self.screen.fill(self.background)

            self.draw_circle('Clique')

            for quadradinho in self.quadradinhos:
                pygame.draw.rect(self.screen, self.cinza, quadradinho)

            pygame.display.update()

        pygame.display.quit()

    def create_quadradinho(self, event):
        if event.button == 1:
            position = pygame.mouse.get_pos()

            if self.button.collidepoint(position):
                rect = self.draw_quadradinho()
                self.quadradinhos.append(rect)
```

```
                self.verifica_colisao(rect)

    def walk_circle(self, event):
        if event.key == pygame.K_a:
            self.button.x -= self.speed

        if event.key == pygame.K_d:
            self.button.x += self.speed

        if event.key == pygame.K_w:
            self.button.y -= self.speed

        if event.key == pygame.K_s:
            self.button.y += self.speed

        for quadradinho in self.quadradinhos:
            if quadradinho.colliderect(self.button):
                self.quadradinhos.remove(quadradinho)

    def draw_circle(self, text):
        pygame.draw.ellipse(self.screen, self.azul, self.button)
        button_text = self.font.render(text, False, self.background)
        button_text_rect =
button_text.get_rect(center=self.button.center)

        self.screen.blit(button_text, button_text_rect)

    def draw_quadradinho(self):
        x = random.randint(25, self.screen.get_width() - 25)
        y = random.randint(25, self.screen.get_height() - 25)

        return pygame.Rect(x, y, 50, 50)

    def verifica_colisao(self, rect):
        for quadrado in self.quadradinhos:
            if quadrado is not rect and quadrado.colliderect(rect):
                self.quadradinhos.remove(quadrado)
                if rect in self.quadradinhos:
                    self.quadradinhos.remove(rect)


game = Collision()
```



10. Obtenha, usando requests ou urllib, dentro de seu programa em Python, o csv do link: https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv
    a. Dentre os seguintes países nórdicos: Suécia, Dinamarca e Noruega, verifique: No século XXI (a partir de 2001), qual foi o maior medalhista de ouro, considerando apenas as seguintes modalidades:
        i. Curling
        ii. Patinação no gelo (skating)
        iii. Esqui (skiing)
        iv. Hóquei sobre o gelo (ice hockey)

```
import pandas as pd


data_frame = pd.read_csv(

'https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv', sep=',')
```

```
filter_paises = data_frame[(data_frame['NOC'] == 'SWE') | (
    data_frame['NOC'] == 'DEN') | (data_frame['NOC'] == 'NOR')]
filter_ano = filter_paises[(filter_paises['Year'] >= 2001)]
filter_esportes = filter_ano[(filter_ano['Sport'] == 'Curling') |
(filter_ano['Sport'] == 'Skating') | (
    filter_ano['Sport'] == 'Skiing') | (filter_ano['Sport'] == 'Ice
Hockey')]

sport_curling = filter_esportes[(filter_esportes['Sport'] == 'Curling')
& (
    filter_esportes['Medal'] == 'Gold')]
curling_result = sport_curling.groupby(['NOC'])['NOC'].count()

print(curling_result)


sport_skating = filter_esportes[(filter_esportes['Sport'] == 'Skating')
& (
    filter_esportes['Medal'] == 'Gold')]
skating_result = sport_skating.groupby(['NOC'])['NOC'].count()

print(skating_result)

sport_skiing = filter_esportes[(filter_esportes['Sport'] == 'Skiing') &
(
    filter_esportes['Medal'] == 'Gold')]
skiing_result = sport_skiing.groupby(['NOC'])['NOC'].count()

print(skiing_result)


sport_ice_hockey = filter_esportes[(filter_esportes['Sport'] == 'Ice
Hockey') & (
    filter_esportes['Medal'] == 'Gold')]
ice_hockeu_result = sport_ice_hockey.groupby(['NOC'])['NOC'].count()

print(ice_hockeu_result)
```

```
PS D:\dev\python\at-py> python3 .\question10_a_winter_olympics.py
NOC
NOR    1
SWE    1
Name: NOC, dtype: int64
Series([], Name: NOC, dtype: int64)
NOC
NOR    10
SWE     4
Name: NOC, dtype: int64
NOC
SWE    1
Name: NOC, dtype: int64
PS D:\dev\python\at-py>
```

b. Para cada esporte, considere todas as modalidades, tanto no masculino quanto no feminino. Sua resposta deve imprimir um relatório mostrando o total de medalhas de cada um dos países e em que esporte, ano, cidade e gênero (masculino ou feminino) cada medalha foi obtida.

```python
import pandas as pd

data_frame = pd.read_csv(

'https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv', sep=',')


filter_bigger_medal = data_frame[(data_frame['NOC'] == 'SWE') | (
    data_frame['NOC'] == 'DEN') | (data_frame['NOC'] == 'NOR')]


filter_ano = filter_bigger_medal[(filter_bigger_medal['Year'] >= 2001)]

filtro_esportes = filter_ano[(filter_ano['Sport'] == 'Curling') |
(filter_ano['Sport'] == 'Skating') | (
    filter_ano['Sport'] == 'Skiing') | (filter_ano['Sport'] == 'Ice Hockey')]

filter_medals_of_gold = filtro_esportes[filtro_esportes['Medal'] == 'Gold']

maior_medalhista_ouro_filtro = filter_medals_of_gold.groupby(
    ['NOC'])['Medal'].count().sort_values()
print(maior_medalhista_ouro_filtro)
```

```
PS D:\dev\python\at-py> python3 .\question10_b_medal_winter_olympics.py
NOC
SWE     6
NOR    11
Name: Medal, dtype: int64
PS D:\dev\python\at-py>
```

11. Obtenha, usando requests ou urllib, dentro de seu programa em Python, o csv do link: https://sites.google.com/site/dr2fundamentospython/arquivos/Video_Games_Sales_as_at_22_Dec_2016.csv. Obtenha, dentre os jogos do gênero de ação (Action), tiro (Shooter) e plataforma (Platform):

    a. Quais são as três marcas que mais publicaram jogos dos três gêneros combinados? Indique também o total de jogos de cada marca.

```python
import pandas as pd

url =
"https://sites.google.com/site/dr2fundamentospython/arquivos/Video_Games
_Sales_as_at_22_Dec_2016.csv"

data_frame = pd.read_csv(url, sep=',')


filter_genre = data_frame[(data_frame['Genre'] == 'Action') | (
    data_frame['Genre'] == 'Shooter') | (data_frame['Genre'] ==
'Platform')]
result_a = filter_genre.groupby(['Publisher'])[
    'Publisher'].count().sort_values(ascending=False).head(3)

print(result_a)
```

```
PS D:\dev\python\at-py> python3 .\games_question_11.py
Publisher
Activision         538
Ubisoft            360
Electronic Arts    344
Name: Publisher, dtype: int64
```

b. Quais são as três marcas que mais venderam os três gêneros combinados? Indique também o total de vendas de cada marca.

```python
import pandas as pd

url =
"https://sites.google.com/site/dr2fundamentospython/arquivos/Video_Games
_Sales_as_at_22_Dec_2016.csv"

data_frame = pd.read_csv(url, sep=',')

# questao a
filter_genre = data_frame[(data_frame['Genre'] == 'Action') | (
    data_frame['Genre'] == 'Shooter') | (data_frame['Genre'] ==
'Platform')]
result_a = filter_genre.groupby(['Publisher'])[
    'Publisher'].count().sort_values(ascending=False).head(3)

# questao b
maior_vendas = data_frame.groupby(
    ['Publisher'])['Global_Sales'].sum().sort_values(ascending=False)
result_b = maior_vendas.head(3)

print(result_b)
```

```
PS D:\dev\python\at-py> python3 .\games_question_11.py
Publisher
Nintendo           1788.81
Electronic Arts    1116.96
Activision          731.16
Name: Global_Sales, dtype: float64
```

c. Qual é a marca com mais publicações em cada um dos gêneros nos últimos dez anos no Japão? Indique também o número total de jogos dela.

```
import pandas as pd

url =
"https://sites.google.com/site/dr2fundamentospython/arquivos/Video_Games
_Sales_as_at_22_Dec_2016.csv"

data_frame = pd.read_csv(url, sep=',')

# questao c
publi_in_japan_action = data_frame[(data_frame['Year_of_Release'] >=
2020 - 10) & (
    data_frame['Genre'] == 'Action')]
group_action = publi_in_japan_action.groupby(['Publisher'])[
    'Publisher'].count().sort_values(ascending=False)

result_c_action = group_action.head(1)

print(result_c_action)

publi_in_japan_shooter = data_frame[(data_frame['Year_of_Release'] >=
2020 - 10) & (
    data_frame['Genre'] == 'Shooter')]
group_shooter = publi_in_japan_shooter.groupby(
    ['Publisher'])['Publisher'].count().sort_values(ascending=False)

result_c_shooter = group_shooter.head(1)

print(result_c_shooter)

publi_in_japan_platform = data_frame[(
    data_frame['Year_of_Release'] >= 2020 - 10) & (data_frame['Genre']
== 'Platform')]
group_platform = publi_in_japan_platform.groupby(
    ['Publisher'])['Publisher'].count().sort_values(ascending=False)

result_c_platform = group_platform.head(1)

print(result_c_platform)
```

```
PS D:\dev\python\at-py> python3 .\games_question_11.py
Publisher
Namco Bandai Games    171
Name: Publisher, dtype: int64
Publisher
Activision    71
Name: Publisher, dtype: int64
Publisher
Nintendo    21
Name: Publisher, dtype: int64
```

d. Qual foi a marca que mais vendeu em cada um desses gêneros nos últimos dez anos, no Japão? Indique também o total de vendas dela.

```python
import pandas as pd

url =
"https://sites.google.com/site/dr2fundamentospython/arquivos/Video_Games
_Sales_as_at_22_Dec_2016.csv"

data_frame = pd.read_csv(url, sep=',')

# questao d
vendas_action_in_action = data_frame[(
    data_frame['Year_of_Release'] >= 2020 - 10) & (data_frame['Genre']
== 'Action')]
group_vendas_action = vendas_action_in_action.groupby(
    ['Publisher'])['JP_Sales'].sum().sort_values(ascending=False)

result_d_action = group_vendas_action.head(1)
print(result_d_action)


vendas_shooter_in_japan = data_frame[(
    data_frame['Year_of_Release'] >= 2020 - 10) & (data_frame['Genre']
== 'Shooter')]
group_vendas_shooter = vendas_shooter_in_japan.groupby(
    ['Publisher'])['JP_Sales'].sum().sort_values(ascending=False)

result_d_shoort = group_vendas_shooter.head(1)
print(result_d_shoort)

vendas_platform_in_japan = data_frame[(
```

```
    data_frame['Year_of_Release'] >= 2020 - 10) & (data_frame['Genre']
== 'Platform')]
group_vendas_platform = vendas_platform_in_japan.groupby(
    ['Publisher'])['JP_Sales'].sum().sort_values(ascending=False)

result_d_platform = group_vendas_platform.head(1)
print(result_d_platform)
```

```
Publisher
Namco Bandai Games    13.11
Name: JP_Sales, dtype: float64
Publisher
Activision    3.85
Name: JP_Sales, dtype: float64
Publisher
Nintendo    14.44
Name: JP_Sales, dtype: float64
PS D:\dev\python\at-py>
```

12. Obtenha, usando requests ou urllib, a página HTML https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html dentro de seu programa em Python e faça:

    a. Imprima o conteúdo referente apenas à tabela apresentada na página indicada.

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

url =
"https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html"
requisicao = requests.get(url)

if requisicao.status_code != 200:
    requisicao.raise_for_status()
else:
    print("Conectado com Sucesso")
```

```
html = requisicao.text
soup = BeautifulSoup(html, "lxml")


def fix_quebra_linha(obj):
    return obj.text.replace('\n', '//')[2:-2].split('//')


rows = []
title = []
for div_table in soup.find_all('div', class_='tabela'):
    for div_titulo in div_table.find_all('div', class_='titulo'):
        title.append(fix_quebra_linha(div_titulo))

        for div_linha in div_table.find_all('div', class_='linha'):
            rows.append(fix_quebra_linha(div_linha))


data_frame = pd.DataFrame(data=rows, columns=title)
result_a = data_frame.to_html(index=False)

print(result_a)
```

```
D:\dev\python\at-py>python3 .\estados_question_12.py
Conectado com Sucesso
<table border="1" class="dataframe">
  <thead>
    <tr>
      <th>Sigla</th>
      <th>Nome</th>
      <th>Capital</th>
      <th>População</th>
      <th>Área</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>DF</td>
      <td>Distrito Federal</td>
      <td>Brasília</td>
      <td>2977216</td>
      <td>5779,999</td>
    </tr>
    <tr>
      <td>GO</td>
      <td>Goiás</td>
      <td>Goiânia</td>
      <td>6730848</td>
      <td>340111,783</td>
    </tr>
    <tr>
      <td>MT</td>
      <td>Mato Grosso</td>
      <td>Cuiabá</td>
      <td>3305531</td>
      <td>903378,292</td>
    </tr>
    <tr>
      <td>MS</td>
      <td>Mato Grosso do Sul</td>
      <td>Campo Grande</td>
      <td>2651235</td>
      <td>357145,532</td>
    </tr>
  </tbody>
</table>
```

b. Escreva um programa que obtenha do usuário uma sigla do estado da região Centro-Oeste e apresenta suas informações correspondentes na tabela. O resultado deve apresentar apenas o conteúdo, sem formatação. Ou seja, as tags não devem aparecer. Não esqueça de checar se a sigla pertence à região.

```python
import requests
from bs4 import BeautifulSoup

url =
"https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html"
requisicao = requests.get(url)
```

```
if requisicao.status_code != 200:
    requisicao.raise_for_status()
else:
    print("Conectado com Sucesso")

html = requisicao.text
soup = BeautifulSoup(html, "lxml")


def fix_quebra_linha(obj):
    return obj.text.replace('\n', '//')[2:-2].split('//')


rows = []
title = []
for div_table in soup.find_all('div', class_='tabela'):
    for div_titulo in div_table.find_all('div', class_='titulo'):
        title.append(fix_quebra_linha(div_titulo))

        for div_linha in div_table.find_all('div', class_='linha'):
            rows.append(fix_quebra_linha(div_linha))


sigla_estado = input("Digite uma região: ")

exist = False
for row in rows:
    if sigla_estado.upper() in row[0]:
        exist = True
        print(row)
        break

if not exist:
    print("Sigla não existe!")
```

```
Digite uma região: go
['GO', 'Goiás', 'Goiânia', '6730848', '340111,783']

D:\dev\python\at-py>
```

13. Obtenha, usando requests ou urllib, o conteúdo sobre as PyLadies no link http://brasil.pyladies.com/about e:

    a. Conte todas as palavras no corpo da página, e indique quais palavras apareceram apenas uma vez.

```
import requests
from bs4 import BeautifulSoup
import re

url = "http://brasil.pyladies.com/about"
requisicao = requests.get(url)

if requisicao.status_code != 200:
    requisicao.raise_for_status()
else:
    print("Conectado com Sucesso")

requisicao.encoding = 'utf-8'

html = requests.get(url).text
soup = BeautifulSoup(html, "lxml")

paragrafos = []
for paragrafo in soup.find_all('p', class_='about-text'):
    paragrafos.append(paragrafo.get_text().replace('\n', "").split())


lists = []
for listas in paragrafos:
    lists += listas

dict = {}
list_just_one_word = []
for word in lists:
    new_string = re.sub(
        u'[^a-zA-Z0-9àáéíóúÁÉÍÓÚâêîôÂÊÎÔãõÃÕçÇ: ]', '', word).lower()

    if new_string in dict:
        dict[new_string] += 1
    else:
        dict[new_string] = 1
        list_just_one_word.append(new_string)

print(f"número de palavras: {len(dict)}")
print(list_just_one_word)
```

R

b. Conte quantas vezes apareceu a palavra ladies no conteúdo da página

```python
import requests
from bs4 import BeautifulSoup
import re

url = "http://brasil.pyladies.com/about"
requisicao = requests.get(url)

if requisicao.status_code != 200:
    requisicao.raise_for_status()
else:
    print("Conectado com Sucesso")

html = requests.get(url).text
soup = BeautifulSoup(html, "lxml")


# questao b
palavra = "ladies"

c = len(re.findall(palavra, soup.get_text()))

print("Ocorrências da palavra", palavra, ":", c)
```

```
Ocorrências da palavra ladies : 4
PS D:\dev\python\at-py>
```