



**ESTI - Escola Superior da Tecnologia da Informação**  
EDC - Graduação em Engenharia de Computação  
Fundamentos de programação com Python  
Assessment

Aluno: Eloy Francisco Barbosa  
Professor: Cassius Figueiredo  
Data: 13/12/2018

Sumário

#01..... 3

#02..... 4

#03..... 4

#04..... 5

#05..... 6

#06..... 8

#07..... 9

#08..... 10

#09..... 12

#10..... 15

    #10.a..... 15

    #10.b. .... 18

#11..... 23

    #11.a..... 23

    #11.b. .... 25

    #11.c..... 25

    #11.d. .... 25

#12..... 26

    #12.a..... 26

    #12.b. .... 28

#13..... 29

    #13.a..... 29

    #13.b. .... 32

#01.

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.

```
print('Nessa atividade vamos criar uma tupla inserindo 3 números
inteiros e vamos apresenta-la na ordem crescente.')
print()
T = ()
i=0

while True:
    try:
        elemento = int(input("Entre com o primeiro elemento: "))
        T+=(elemento,)
    except ValueError:
        print("Favor digitar um número inteiro")
    else:
        break

while True:
    try:
        while i < 2:

            elemento = int(input("Entre com o próximo elemento:
"))
            i+=1
            T+=(elemento,)
        except ValueError:
            print("Favor digitar um número inteiro")

    else:
        break

T_crescente=(sorted(T))

print('\nA tupla criada na ordem crescente ficou da seguinte
forma: ', T_crescente)
```

Shell

```
>>> %Run 01.py
```

```
Nessa atividade vamos criar uma tupla inserindo 3 números inteiros e vamos apresenta-la na ordem crescente.
```

```
Entre com o primeiro elemento: 2
```

```
Entre com o próximo elemento: 3
```

```
Entre com o próximo elemento: 4
```

```
A tupla criada na ordem crescente ficou da seguinte forma: [2, 3, 4]
```

```
>>> |
```

## #02.

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.

```
print('Nessa atividade vamos somar todos os números pares de 1
ate N')

n=int(input("Insira o valor de N:"))
print('Agora vamos somar todos os numeros pares de 1 até', n)

resultado = 0
contador = 0

for c in range(0, n+1, 2):
    print(c, end=' ')
    resultado = resultado + c
    contador = contador + 1

print()
print('No intervalo entre 1 e',n,'temos',contador-1,'números
pares, e a soma de todos os números resulta em',resultado)
```

Shell

>>> %Run 02.py

Nessa atividade vamos somar todos os números pares de 1 ate N

Insira o valor de N:30

Agora vamos somar todos os numeros pares de 1 até 30

0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

No intervalo entre 1 e 30 temos 15 números pares, e a soma de todos os números resulta em 240

>>> |

## #03.

Usando o Thonny, escreva uma função em Python chamada potencia. 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.

```
def potencia (a, b):
    R = a
    for i in range (1, b):
        R*=a
    return R

print('Nesta atividade vamos informar dois números inteiros "A"
e "B" e a partir desses números vamos calcular o valor de A
elevado B. (Favor limitar-se a números menores que 100)')
a=int(input('Insira o valor de A:'))
b=int(input('Insira o valor de B:'))

print('O resultado de', a, 'elevado a', b,'é:', potencia (a, b))
```

```
Shell
>>> %Run 03.py
Nesta atividade vamos informar dois números inteiros "A" e "B" e a partir desses números vamos calcular o valor de A elevado B. (Favor limitar-se a números menores que 100)
Insira o valor de A:2
Insira o valor de B:10
O resultado de 2 elevado a 10 é: 1024
>>> |
```

#### #04.

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. Exemplo: se a entrada for [4, 3, 5, 1, 2], o resultado deve ser [2, 1, 5, 3, 4].

```
print('Nessa atividade vamos criar uma lista inserindo 5 números inteiros e vamos apresentá-la na ordem inversa de inserção.')
print()
L = []
T=len(L)
i=0

while True:
    try:
        elemento = int(input("Entre com o primeiro elemento: "))
        L+=[elemento]
    except ValueError:
        print("Favor digitar um número inteiro")
    else:
        break

while True:
    try:
        while i < 4:

            elemento = int(input("Entre com o próximo elemento: "))

            i+=1
            L+=[elemento]
        except ValueError:
            print("Favor digitar um número inteiro")

        else:
            break

L_reversa=L[::-1]

print('\nA lista criada na ordem inversa fica da seguinte forma: ', L_reversa)
```

```
Shell
Python 3.6.6
>>> %Run 04.py

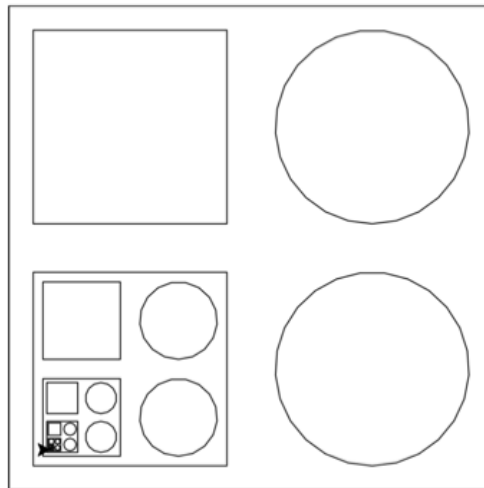
Nessa atividade vamos criar uma lista inserindo 5 números inteiros e vamos apresenta-la na ordem
inversa de inserção.

Entre com o primeiro elemento: 2
Entre com o próximo elemento: 3
Entre com o próximo elemento: 4
Entre com o próximo elemento: 5
Entre com o próximo elemento: 6

A lista criada na ordem inversa fica da seguinte forma: [6, 5, 4, 3, 2]
>>> |
```

### #05.

Usando a biblioteca 'turtle' crie uma função que desenhe a imagem a seguir:



```
import turtle

tamanho = 600
margem = 40
X = -300
Y = -300

def quadrado(x, y):
    turtle.shape('turtle')
    turtle.speed(100)
    turtle.penup()
    turtle.setx(x + margem)
    turtle.sety(y + margem)

    turtle.pendown()
    turtle.setx(turtle.xcor() + tamanho - 2 * margem)
    turtle.sety(turtle.ycor() + tamanho - 2 * margem)

    turtle.setx(turtle.xcor() - tamanho + 2 * margem)
    turtle.sety(turtle.ycor() - tamanho + 2 * margem)
```

```

def circulo(x, y):
    turtle.shape('turtle')
    turtle.speed(100)
    turtle.penup()
    turtle.setx(x + margem + (tamanho - margem * 2) / 2)
    turtle.sety(y + margem)

    turtle.pendown()
    turtle.circle((tamanho - margem * 2) / 2)

quadrado(X, Y)

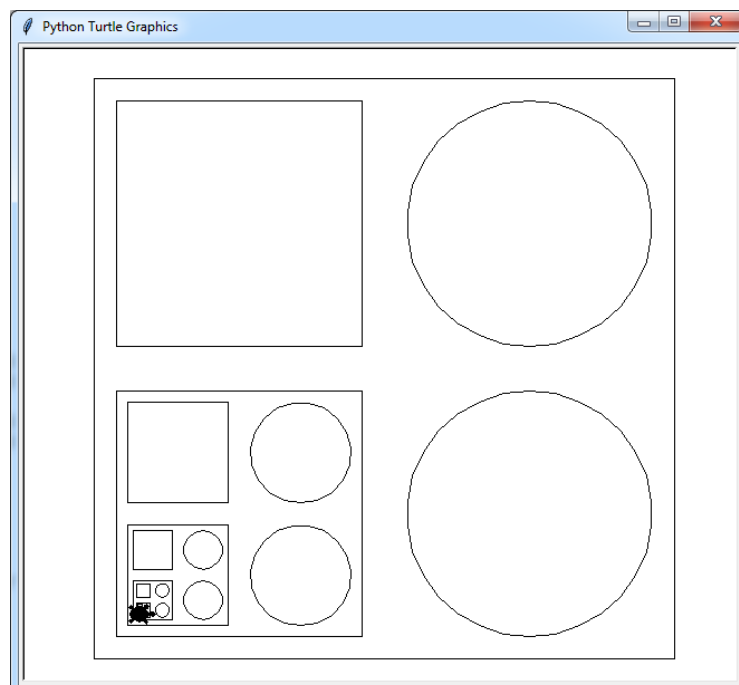
for i in range(8):

    tamanho = tamanho / 2 - margem
    X += margem
    Y += margem
    margem /= 2

    quadrado(X, Y)
    quadrado(X, Y + tamanho)
    circulo(X + tamanho, Y)
    circulo(X + tamanho, Y + tamanho)

turtle.done()

```



#06.

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.

```
print('Nesta atividade vamos criar uma Tupla contendo 5 números INTEIROS, depois vamos retornar uma lista contendo somente os números ímpares e uma Nova Tupla contendo somente os elementos nas posições pares.')
print()
print('Lembrando que ao retornar os elementos nas posições pares iremos considerar o índice da Tupla criada que inicia em "0"
OBS.: Nessa atividade o "0" será considerado como par.')
print()
tupla = ()
lista_impar = []
i = 1
tupla_par = ()

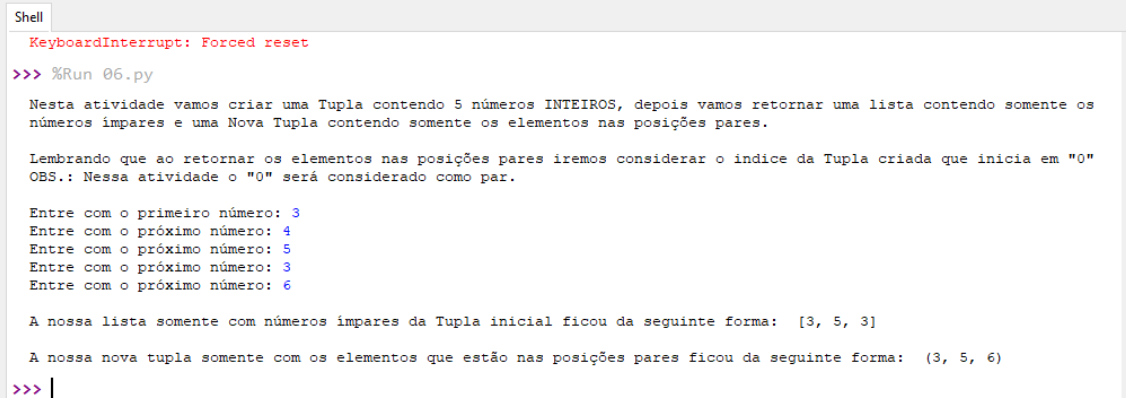
n = int(input("Entre com o primeiro número: "))
tupla += (n,)
for c in range(1, 5):

    n = int(input("Entre com o próximo número: "))
    tupla += (n,)
    c += 1

tamanho_tupla = len(tupla)

for p in range(0, tamanho_tupla, 2):
    tupla_par += (tupla[p],)

for i in range(0, tamanho_tupla):
    if tupla[i] % 2 != 0:
        lista_impar.append(tupla[i])
print()
print('A nossa lista somente com números ímpares da Tupla inicial ficou da seguinte forma: ', lista_impar)
print()
print('A nossa nova tupla somente com os elementos que estão nas posições pares ficou da seguinte forma: ', tupla_par)
```



```
Shell
KeyboardInterrupt: Forced reset

>>> %Run 06.py

Nesta atividade vamos criar uma Tupla contendo 5 números INTEIROS, depois vamos retornar uma lista contendo somente os números ímpares e uma Nova Tupla contendo somente os elementos nas posições pares.

Lembrando que ao retornar os elementos nas posições pares iremos considerar o índice da Tupla criada que inicia em "0"
OBS.: Nessa atividade o "0" será considerado como par.

Entre com o primeiro número: 3
Entre com o próximo número: 4
Entre com o próximo número: 5
Entre com o próximo número: 3
Entre com o próximo número: 6

A nossa lista somente com números ímpares da Tupla inicial ficou da seguinte forma: [3, 5, 3]

A nossa nova tupla somente com os elementos que estão nas posições pares ficou da seguinte forma: (3, 5, 6)

>>> |
```



#07.

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

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)
amarelo = (255,255,0)

pygame.init()
tela = pygame.display.set_mode([640, 480])
pygame.display.set_caption("Atividade 07")
relogio = pygame.time.Clock()
tela.fill(preto)
terminou = False

def quadrado_amarelo():
    x = random.randint(25, 615)
    y = random.randint(25, 455)
    pygame.draw.rect(tela, amarelo, (x, y, 50, 50))

while not terminou:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            terminou = True

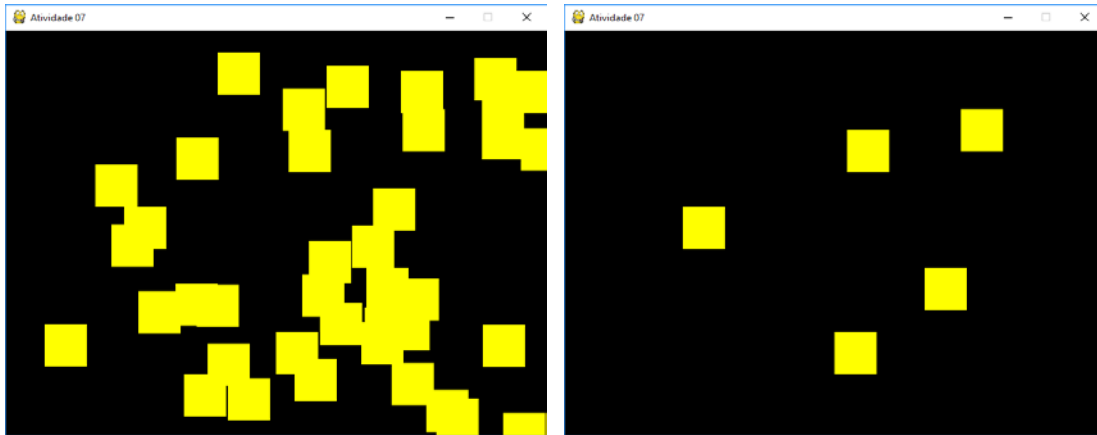
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_SPACE:
                quadrado_amarelo()

        if event.type == pygame.MOUSEBUTTONDOWN:
            quadrado_amarelo()

    pygame.display.update()
    relogio.tick(27)

pygame.display.quit()

pygame.quit()
```



#### #08.

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 random
import math

largura = 640
altura = 480
branco = (255,255,255)
preto = (0, 0, 0)
vermelho= (155, 0, 0)
azul = (0, 255, 255)
verde = (0, 255, 0)
amarelo = (255, 255, 0)

pygame.init()
pygame.display.set_caption('Atividade 08')

fonte = pygame.font.SysFont('Courier ', 21)
tela = pygame.display.set_mode((largura, altura))
clock = pygame.time.Clock()
fim = False

quadrados = []

circulo = {"x": 320, "y": 60, "raio": 50}

def circulo_vermelho():
    pygame.draw.circle(tela, vermelho, (circulo["x"],
circulo["y"]), circulo['raio'])
    textsurface = fonte.render("CLIQUE", False, branco)
    tela.blit(textsurface, (292, 52))
```

```

def retangulo_amarelo(pos):
    dist = math.sqrt((circulo["x"] - pos[0])**2 + (circulo["y"]
- pos[1])**2)
    if dist > circulo["raio"]:
        return

    x = random.randint(0, largura - 100)
    y = random.randint(0, altura - 50)

    for quadrado in quadrados:
        rect1 = pygame.Rect((quadrado[0], quadrado[1], 100, 50))
        rect2 = pygame.Rect((x, y, 100, 50))

        if rect1.colliderect(rect2):
            quadrados.remove(quadrado)
            return

    quadrados.append((x, y))

while not fim:

    clock.tick(30)

    tela.fill(preto)

    for x, y in quadrados:
        pygame.draw.rect(tela, amarelo, (x, y, 100, 50))

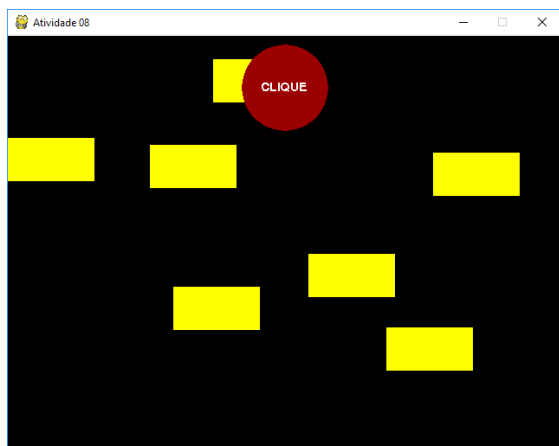
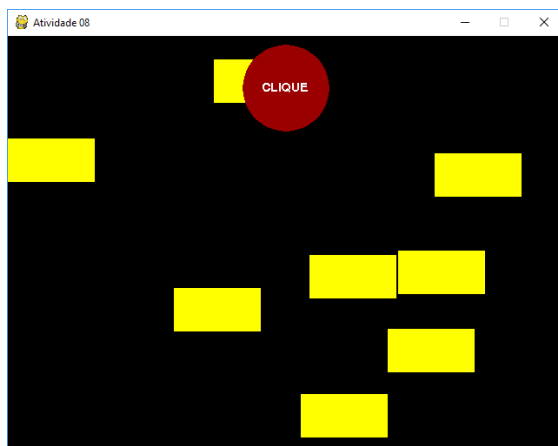
    circulo_vermelho()

    pygame.display.update()

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            fim = True
            exit(0)
            break

        if event.type == pygame.MOUSEBUTTONDOWN:
            retangulo_amarelo(pygame.mouse.get_pos())

```



**#09.**

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 random
import math

largura = 640
altura = 480

branco = (255, 255, 255)
preto = (0, 0, 0)
vermelho = (155, 0, 0)
azul = (0, 255, 255)
verde = (0, 255, 0)
amarelo = (255, 255, 0)

pygame.init()
pygame.display.set_caption('9')

fonte = pygame.font.SysFont('Courier New', 21)
tela = pygame.display.set_mode((largura, altura))
clock = pygame.time.Clock()
fim = False

quadrados = []

circulo = {"x": 60, "y": 60, "raio": 50, "dir": None, "speed": 10}

def circulo_vermelho():
    pygame.draw.circle(tela, vermelho, (circulo["x"],
    circulo["y"]), circulo['raio'])
    textsurface = fonte.render("Clique", False, branco)
    tela.blit(textsurface, (circulo["x"] - 25, circulo["y"] - 10))

def retangulo_amarelo(pos):
    dist = math.sqrt((circulo["x"] - pos[0])**2 + (circulo["y"] - pos[1])**2)
    if dist > circulo["raio"]:
        return

    x = random.randint(0, largura - 100)
    y = random.randint(0, altura - 50)

    for quadrado in quadrados:
        rect1 = pygame.Rect((quadrado[0], quadrado[1], 100, 50))
        rect2 = pygame.Rect((x, y, 100, 50))

        if rect1.colliderect(rect2):
            quadrados.remove(quadrado)
```

```

        return

    quadrados.append((x, y))

def mover_circulo():
    if circulo["dir"] is None:
        return

    circulo["x"] += int(math.cos(math.radians(circulo["dir"])) *
circulo["speed"])
    circulo["y"] -= int(math.sin(math.radians(circulo["dir"])) *
circulo["speed"])

    for quadrado in quadrados:
        if checa_colisao((quadrado[0], quadrado[1], 100, 50),
(circulo["x"], circulo["y"], circulo["raio"])):
            quadrados.remove(quadrado)

def checa_colisao(rect, circle):
    def dentro_circle(ponto):
        dist = math.sqrt((circle[0] - ponto[0]) ** 2 +
(circle[1] - ponto[1]) ** 2)
        if dist <= circle[2]:
            return True, dist
        return False, dist

    def dentro_rect(ponto):
        if ((rect[0] <= ponto[0] <= rect[0] + rect[2]) and
            (rect[1] <= ponto[1] <= rect[1] + rect[3])):
            return True
        return False

    pontos_rect = ((rect[0], rect[1]),
                    (rect[0] + rect[2], rect[1]),
                    (rect[0], rect[1] + rect[3]),
                    (rect[0] + rect[2], rect[1] + rect[3]))

    pontos_circle = ((circle[0] + circle[2], circle[1]),
                     (circle[0], circle[1] + circle[2]),
                     (circle[0] - circle[2], circle[1]),
                     (circle[0], circle[1] - circle[2]))

    # distancia minima para alguma aresta encostar, maior q isso
    esta
    # longe o suficiente para ignorar
    minima_dist = circle[2]*2 + (rect[2] if rect[2] > rect[3]
else rect[3])

    for ponto_circle in pontos_circle:
        if dentro_rect(ponto_circle):
            return True

    for ponto_rect in pontos_rect:
        result = dentro_circle(ponto_rect)
        if result[0]:
            return True

```

```

        elif result[1] > minima_dist:
            return False

    return False

while not fim:
    # configurando o clock para 30 vezes por segundo
    clock.tick(30)

    tela.fill(preto)

    for x, y in quadrados:
        pygame.draw.rect(tela, amarelo, (x, y, 100, 50))

    circulo_vermelho()
    mover_circulo()

    pygame.display.update()

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            fim = True
            exit(0)
            break

        if event.type == pygame.MOUSEBUTTONDOWN:
            retangulo_amarelo(pygame.mouse.get_pos())

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_w:
                circulo["dir"] = 90

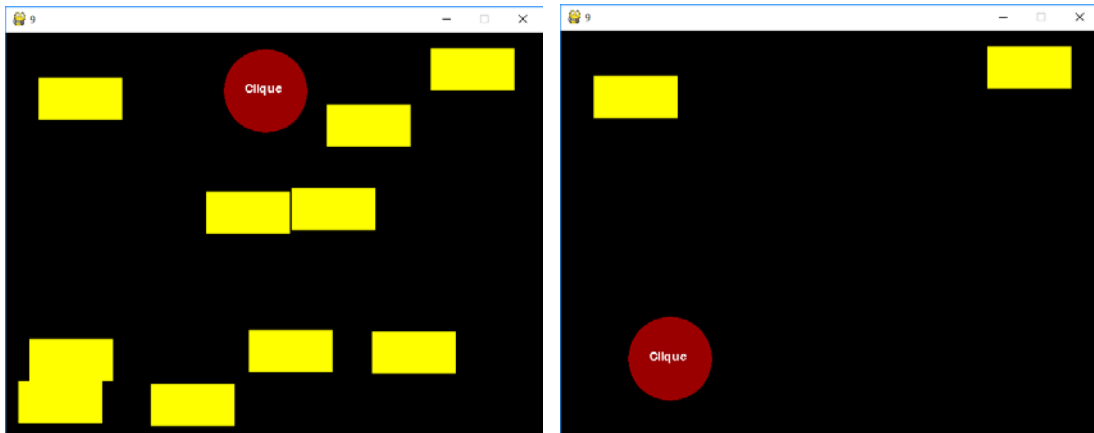
            elif event.key == pygame.K_s:
                circulo["dir"] = 270

            elif event.key == pygame.K_a:
                circulo["dir"] = 180

            elif event.key == pygame.K_d:
                circulo["dir"] = 0

        if event.type == pygame.KEYUP:
            circulo["dir"] = None

```



#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](https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv) E:

#10.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 requests
import re
from bs4 import BeautifulSoup

print('Nesta atividade vamos utilizar o requests para obter o
conteúdo do seguinte arquivo CSV:
https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv')

print('\nE depois vamos exibir dentre os tres países nórdicos:
Suécia, Dinamarca e Noruega , a partir do Século XXI, qual foi o
maior medalhista considerando apenas as seguintes modalidades:
Curling, Skatin, Skiing e Ice Hockey')
url =
'https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv'

conn = requests.get(url, timeout=5)

if conn.status_code != 200:
    conn.raise_for_status()
else:
    print("Conectado com sucesso!")

csv = requests.get(url).text
```

```

csv_lista = csv.splitlines()
tam_csv_lista = len(csv_lista)

lista=[]

for i in range(1, tam_csv_lista):
    temp = csv_lista[i].split(',')
    lista.append(temp[:])
    temp.clear()

tam_lista = len(lista)

lista_ano=[]

for i in range (0, tam_lista):
    temp = lista[i]
    if lista[i][0] == '2001' or lista[i][0] == '2002' or
lista[i][0] == '2003' or lista[i][0] == '2004' or lista[i][0] ==
'2005' or lista[i][0] == '2006':
        lista_ano.append(temp[:])
        temp.clear()

tam_lista_ano= len(lista_ano)

nordicos = [] #noruega, suecia, dinamarca

for i in range (0, tam_lista_ano):
    temp = lista_ano[i]
    if lista_ano[i][4] == 'NOR':
        nordicos.append(temp[:])
        temp.clear()

    elif lista_ano[i][4] == 'SWE':
        nordicos.append(temp[:])
        temp.clear()

    elif lista_ano[i][4] == 'DEN':
        nordicos.append(temp[:])
        temp.clear()

tam_nordicos = len(nordicos)

gold = []

for i in range (0, tam_nordicos):
    temp = nordicos[i]
    if nordicos[i][7] == 'Gold':
        gold.append(temp[:])
        temp.clear()

tam_gold = len(gold)

esportes = []

for i in range (0, tam_gold):
    temp = gold[i]

```



```

    if gold[i][2] == 'Curling':
        esportes.append(temp[:])
        temp.clear()

    elif gold[i][2] == 'Skating':
        esportes.append(temp[:])
        temp.clear()

    elif gold[i][2] == 'Skiing':
        esportes.append(temp[:])
        temp.clear()

    elif gold[i][2] == 'Ice Hockey':
        esportes.append(temp[:])
        temp.clear()

tam_esportes = len(esportes)

nor = 0
swe = 0
den = 0

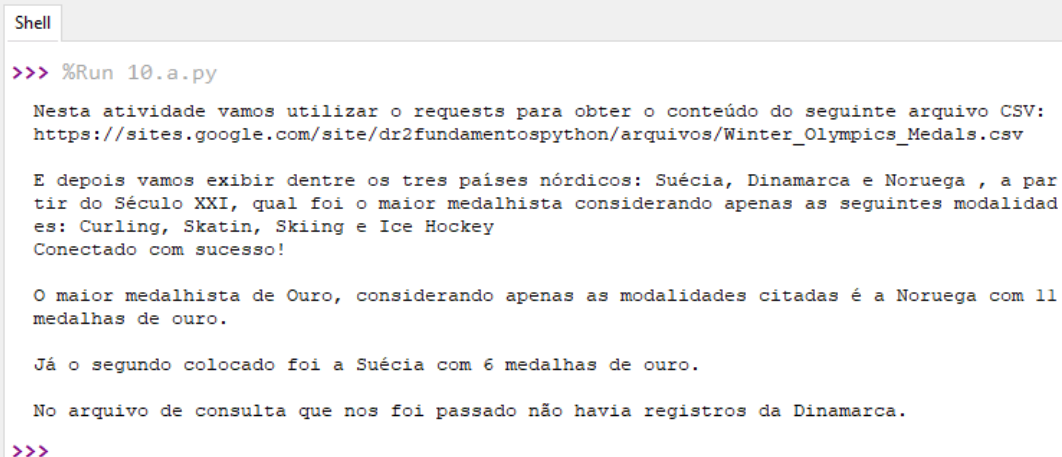
for i in range (0, tam_esportes):
    if esportes[i][4] == 'NOR':
        nor += 1

for i in range (0, tam_esportes):
    if esportes[i][4] == 'SWE':
        swe += 1

for i in range (0, tam_esportes):
    if esportes [i][4] == 'DEN':
        den += 1

print(f'\nO maior medalhista de Ouro, considerando apenas as
modalidades citadas é a Noruega com {nor} medalhas de ouro.')
print(f'\nJá o segundo colocado foi a Suécia com {swe} medalhas
de ouro.' )
print(f'\nNo arquivo de consulta que nos foi passado não havia
registros da Dinamarca.')

```



```

Shell

>>> %Run 10.a.py

Nesta atividade vamos utilizar o requests para obter o conteúdo do seguinte arquivo CSV:
https://sites.google.com/site/dr2fundamentospython/arquivos/Winter_Olympics_Medals.csv

E depois vamos exibir dentre os tres países nórdicos: Suécia, Dinamarca e Noruega , a par
tir do Século XXI, qual foi o maior medalhista considerando apenas as seguintes modalidad
es: Curling, Skatin, Skiing e Ice Hockey
Conectado com sucesso!

O maior medalhista de Ouro, considerando apenas as modalidades citadas é a Noruega com 11
medalhas de ouro.

Já o segundo colocado foi a Suécia com 6 medalhas de ouro.

No arquivo de consulta que nos foi passado não havia registros da Dinamarca.

>>>

```

#### #10.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.

```
import requests
import json
from collections import Counter

url =
"https://sites.google.com/site/dr2fundamentospython/arquivos/Win
ter_Olympics_Medals.csv"
conn = requests.Session().get(url)

if conn.status_code != 200:
    conn.raise_for_status()
else:
    print("Conectado com sucesso!")

rows = [data.split(",") for data in conn.text.split("\n")]

header = rows.pop(0)
lister = []

for row in rows:
    lister.append({header[index]: item for index, item in
enumerate(row)})

def relatorio(json_Format=False):
    CountMedal = dict(Counter([item["NOC"] for item in lister]))

    talkative = {}

    for key in CountMedal.keys():
        talkative[key] = {}
        talkative[key]["Total de Medalhas"] = CountMedal[key]

    for key in talkative.keys():
        talkative[key]["Medalhas"] = []

        for medal in [item for item in lister if item["NOC"] ==
key]:
            medalRows = {"Esporte": medal["Sport"], "Ano":
medal["Year"], "Cidade": medal["City"], "Genero": "Masculino" if
medal["Event gender"] == "M" else "Femenino"}

            talkative[key]["Medalhas"].append(medalRows)

    if json_Format:
        print(json.dumps(talkative, indent=1))

    else:
        for country in talkative.keys():
```

```

        print("\n")
        print('Total de Medalhas: %d' %
talkative[country]['Total de Medalhas'])

        defaultForms = "{:<14}{:<6}{:<10}{:<22}"
        print(defaultForms.format('Esporte', 'Ano',
'Genero', 'Cidade', 'Pais'))
        for medals in talkative[country]['Medalhas']:
            print(defaultForms.format(medals['Esporte'],
medals['Ano'], medals['Genero'], medals['Cidade']))

print('\nNesta atividade vamos exibir um relatório mostrando o
total de medalhas de cada um dos países dividido por esporte,
ano, cidade e gênero.\n')

print('\n===== RELATÓRIO =====')

relatorio()

```



```

Shell
>>> %Run 10.b.py

Conectado com sucesso!

Nesta atividade vamos exibir um relatório mostrando o total de medalhas de cada um dos pa
íses dividido por esporte, ano, cidade e gênero.

===== RELATÓRIO =====

Total de Medalhas: 185
Esporte      Ano  Genero  Cidade
Skiing       2006 Masculino Turin

Total de Medalhas: 5
Esporte      Ano  Genero  Cidade
Skating      1998 Masculino Nagano

Total de Medalhas: 119
Esporte      Ano  Genero  Cidade
Skiing       2006 Femenino Turin

Total de Medalhas: 151
Esporte      Ano  Genero  Cidade
Skiing       2006 Masculino Turin

```

Shell				
Total de Medalhas: 83				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Masculino	Turin	
Total de Medalhas: 21				
Esporte	Ano	Genero	Cidade	
Bobsleigh	2006	Femenino	Turin	
Total de Medalhas: 280				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 118				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 118				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 216				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 158				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Shell				
Total de Medalhas: 25				
Esporte	Ano	Genero	Cidade	
Skiing	1992	Masculino	Albertville	
Total de Medalhas: 6				
Esporte	Ano	Genero	Cidade	
Skating	1980	Femenino	Lake Placid	
Total de Medalhas: 101				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Masculino	Turin	
Total de Medalhas: 41				
Esporte	Ano	Genero	Cidade	
Skiing	1988	Masculino	Calgary	
Total de Medalhas: 78				
Esporte	Ano	Genero	Cidade	
Skating	2006	Masculino	Turin	
Total de Medalhas: 19				
Esporte	Ano	Genero	Cidade	
Skiing	1964	Masculino	Innsbruck	
Total de Medalhas: 32				
Esporte	Ano	Genero	Cidade	
Skating	2006	Femenino	Turin	

Shell				
Total de Medalhas: 8				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 194				
Esporte	Ano	Genero	Cidade	
Skiing	1988	Masculino	Calgary	
Total de Medalhas: 2				
Esporte	Ano	Genero	Cidade	
Skating	1992	Femenino	Albertville	
Total de Medalhas: 110				
Esporte	Ano	Genero	Cidade	
Skating	1988	Femenino	Calgary	
Total de Medalhas: 1				
Esporte	Ano	Genero	Cidade	
Bobsleigh	1968	Masculino	Grenoble	
Total de Medalhas: 2				
Esporte	Ano	Genero	Cidade	
Skiing	1992	Femenino	Albertville	
Total de Medalhas: 9				
Esporte	Ano	Genero	Cidade	
Skiing	1988	Masculino	Calgary	
Shell				
Total de Medalhas: 6				
Esporte	Ano	Genero	Cidade	
Skating	2006	Femenino	Turin	
Total de Medalhas: 4				
Esporte	Ano	Genero	Cidade	
Skiing	1988	Masculino	Calgary	
Total de Medalhas: 33				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 23				
Esporte	Ano	Genero	Cidade	
Skiing	1992	Femenino	Albertville	
Total de Medalhas: 31				
Esporte	Ano	Genero	Cidade	
Skating	2006	Masculino	Turin	
Total de Medalhas: 2				
Esporte	Ano	Genero	Cidade	
Skiing	1992	Masculino	Albertville	
Total de Medalhas: 1				
Esporte	Ano	Genero	Cidade	
Skiing	1992	Femenino	Albertville	

Shell				
Total de Medalhas: 6				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Masculino	Turin	
Total de Medalhas: 6				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Masculino	Turin	
Total de Medalhas: 5				
Esporte	Ano	Genero	Cidade	
Skiing	1998	Masculino	Nagano	
Total de Medalhas: 76				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Masculino	Turin	
Total de Medalhas: 4				
Esporte	Ano	Genero	Cidade	
Skiing	2002	Masculino	Salt Lake City	
Total de Medalhas: 5				
Esporte	Ano	Genero	Cidade	
Skating	2006	Femenino	Turin	
Total de Medalhas: 1				
Esporte	Ano	Genero	Cidade	
Skiing	1994	Femenino	Lillehammer	
Shell				
Esporte	Ano	Genero	Cidade	
Skiing	1994	Femenino	Lillehammer	
Total de Medalhas: 10				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 1				
Esporte	Ano	Genero	Cidade	
Curling	1998	Femenino	Nagano	
Total de Medalhas: 7				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 6				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Femenino	Turin	
Total de Medalhas: 1				
Esporte	Ano	Genero	Cidade	
Luge	2006	Masculino	Turin	
Total de Medalhas: 1				
Esporte	Ano	Genero	Cidade	
Skiing	2006	Masculino	Turin	
>>>				

#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](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):

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

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

#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.

#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.

#11.a.

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

```
import requests
import re
from bs4 import BeautifulSoup
from collections import Counter

print('Nesta atividade vamos utilizar o requests para obter o
conteúdo do seguinte arquivo CSV:
https://sites.google.com/site/dr2fundamentospython/arquivos/Video\_Games\_Sales\_as\_at\_22\_Dec\_2016.csv')

print('\nE depois vamos obter informações somente dos seguintes
generos de jogos:ação (Action), tiro (Shooter) e plataforma
(Platform):')

print('\nE dpor fim informar qual as três marcas que mais
publicaram jogos dos 3 generos combinados e o total de jogo de
cada marca.')
```

```
url =
'https://sites.google.com/site/dr2fundamentospython/arquivos/Video\_Games\_Sales\_as\_at\_22\_Dec\_2016.csv'

conn = requests.get(url, timeout=5)

if conn.status_code != 200:
    conn.raise_for_status()
else:
    print("Conectado com sucesso!")

csv = requests.get(url).text
```

```

csv_lista = csv.splitlines()

tam_csv_lista = len(csv_lista)

lista=[]

for i in range(1, tam_csv_lista):
    temp = csv_lista[i].split(',')
    lista.append(temp[:])
    temp.clear()

tam_lista = len(lista)

generos = []

for i in range (0, tam_lista):
    temp = lista[i]
    if lista[i][3] == 'Action':
        generos.append(temp[:])

    elif lista[i][3] == 'Shooter':
        generos.append(temp[:])

    elif lista[i][3] == 'Platform':
        generos.append(temp[:])
    temp.clear()

tam_generos = len(generos)

lista_limpa=[]

for i in range (tam_generos):
    del(generos[i][10:])

total=[]

for i in range (tam_generos):
    temp=generos[i]
    total.append(temp[4])
    temp.clear()

marcas=sorted(set(total))

marcas_total=[]

for i in range(len(marcas)):
    temp=marcas[i]
    temp2=total.count(temp)
    marcas_total.append(temp2)

tresmaiores=sorted(marcas_total, reverse=True)
tresmaiores=tresmaiores[:3]

primeirolugar=marcas_total.index(tresmaiores[0])
primeirolugar=marcas[primeirolugar]

```



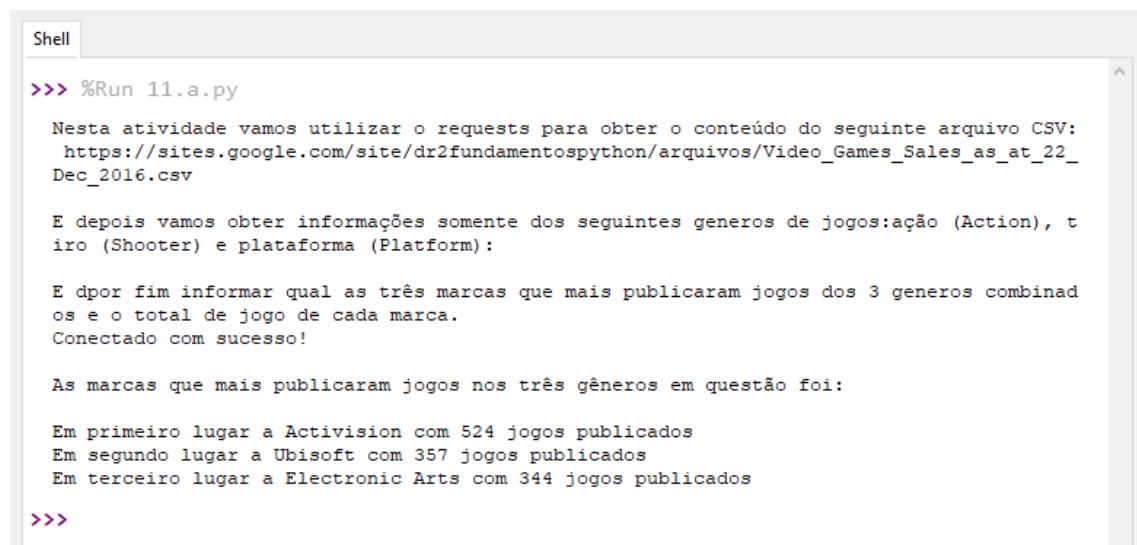
```

segundolugar=marcas_total.index(tresmaiores[1])
segundolugar=marcas[segundolugar]

terceirolugar=marcas_total.index(tresmaiores[2])
terceirolugar=marcas[terceirolugar]

print('\nAs marcas que mais publicaram jogos nos três gêneros em
questão foi:\n')
print(f'Em primeiro lugar a {primeirolugar} com {tresmaiores[0]}
jogos publicados')
print(f'Em segundo lugar a {segundolugar} com {tresmaiores[1]}
jogos publicados')
print(f'Em terceiro lugar a {terceirolugar} com {tresmaiores[2]}
jogos publicados')

```



```

Shell

>>> %Run 11.a.py

Nesta atividade vamos utilizar o requests para obter o conteúdo do seguinte arquivo CSV:
https://sites.google.com/site/dr2fundamentospython/arquivos/Video_Games_Sales_as_at_22_
Dec_2016.csv

E depois vamos obter informações somente dos seguintes generos de jogos:ação (Action), t
iro (Shooter) e plataforma (Platform):

E dpor fim informar qual as três marcas que mais publicaram jogos dos 3 generos combinad
os e o total de jogo de cada marca.
Conectado com sucesso!

As marcas que mais publicaram jogos nos três gêneros em questão foi:

Em primeiro lugar a Activision com 524 jogos publicados
Em segundo lugar a Ubisoft com 357 jogos publicados
Em terceiro lugar a Electronic Arts com 344 jogos publicados

>>>

```

**#11.b.**

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

Não Consegui fazer a tempo.

**#11.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.**

Não Consegui fazer a tempo.

**#11.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.**

Não Consegui fazer a tempo.

#12.

Obtenha, usando requests ou urllib, a página HTML [https://fgopassos.github.io/pagina\\_exemplo/estadosCentroOeste.html](https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html) dentro de seu programa em Python e faça:

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

#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.

#12.a.

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

```
import requests
from bs4 import BeautifulSoup

print('Nesta atividade vamos obter dados utilizando o requests
da seguinte página da web:
https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html')
print()
print('E depois de obter os dados vamos exibir apenas o conteúdo
referente à tabela apresentada na pagina indicada')
lista=[]
texto=''

url =
"https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html"

html = requests.get(url).text

soup = BeautifulSoup(html,"lxml")

for i in soup.html.find_all('article'):
    texto += i.text
print ()
print('Abaixo segue o apenas o conteúdo da tabela, agrupado
linha por linha', texto)
```

```
Shell
>>> %Run 12.a.py

Nesta atividade vamos obter dados utilizando o requests da seguinte página da web:
https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html

E depois de obter os dados vamos exibir apenas o conteúdo referente à tabela apres
entada na página indicada

Abaixo segue o apenas o conteúdo da tabela, agrupado linha por linha

Sigla
Nome
Capital
População
Área

DF
Distrito Federal
Brasília
2977216
5779,999

GO
Goiás
Goiânia
6730848
340111,783

MT
Mato Grosso
Cuiabá
3305531
903378,292

MS
Mato Grosso do Sul
Campo Grande
2651235
357145,532

>>>
```

### #12.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.

```
import requests
from bs4 import BeautifulSoup

print('Nesta atividade vamos obter dados utilizando o requests da seguinte página da web: https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html')
print()
print('E depois o usuário vai inserir uma sigla de um estado da região Centro-Oeste e com isso vamos apresentar as informações contidas na tabela sobre esse estado.')
print()
texto=''

url = "https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html"

html = requests.get(url).text

soup = BeautifulSoup(html,"lxml")

for i in soup.html.find_all('article'):
    texto += i.text
    lista = (texto.splitlines())

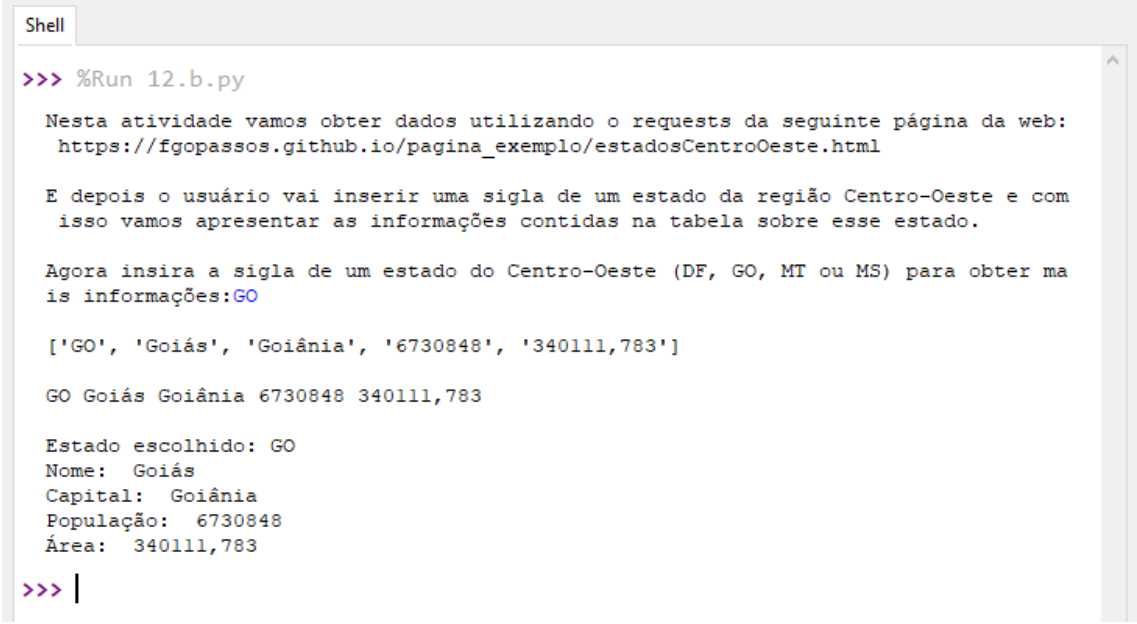
sigla = input('Agora insira a sigla de um estado do Centro-Oeste (DF, GO, MT ou MS) para obter mais informações:')
if sigla not in lista:
    print()
    print('A sigla inserida não corresponde a um estado do Centro-Oeste')
else:
    if sigla=='DF' or sigla=='GO' or sigla=='MT' or sigla=='MS':
        resultado =
        lista[(lista.index(sigla)):(lista.index(sigla)+5)]

        print()
        print(resultado)
        print()
        print(' '.join(resultado))
        print()
        print('Estado escolhido:', resultado[0],
              '\nNome: ', resultado[1],
              '\nCapital: ', resultado[2],
              '\nPopulação: ', resultado[3],
              '\nÁrea: ', resultado[4])
    else:
```

```

        print()
        print('A sigla inserida não corresponde a um estado do
Centro-Oeste')

```



```

Shell

>>> %Run 12.b.py

Nesta atividade vamos obter dados utilizando o requests da seguinte página da web:
https://fgopassos.github.io/pagina_exemplo/estadosCentroOeste.html

E depois o usuário vai inserir uma sigla de um estado da região Centro-Oeste e com
isso vamos apresentar as informações contidas na tabela sobre esse estado.

Agora insira a sigla de um estado do Centro-Oeste (DF, GO, MT ou MS) para obter ma
is informações:GO

['GO', 'Goiás', 'Goiânia', '6730848', '340111,783']

GO Goiás Goiânia 6730848 340111,783

Estado escolhido: GO
Nome: Goiás
Capital: Goiânia
População: 6730848
Área: 340111,783

>>> |

```

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

**#13.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
from collections import Counter

print('Nesta atividade vamos utilizar o requests para obter o
conteúdo da seguinte página da Web:
http://brasil.pyladies.com/about')

print('E depois vamos informar quantas palavra existem no corpo
da página, informar quantas aparecem somente uma vez e por fim
vamos exibir todas as palavras que aparecem somenteo uma vez.')
print()

lista=[]

umavez=[]

texto=' '

```

Shell

```
>>> %Run 13.a.py
```

Nesta atividade vamos utilizar o requests para obter o conteúdo da seguinte página da Web: <http://brasil.pyladies.com/about>  
E depois vamos informar quantas palavra existem no corpo da página, informar quantas aparecem somente uma vez e por fim vamos exibir todas as palavras que aparecem somente uma vez.

No corpo da página existem no total 208 palavras e 116 aparecem somente uma vez, segue abaixo a relação das palavras que aparecem somente uma vez:

O  
PyLadies  
comunidade  
mundial  
foi  
trazida  
ao  
Brasil  
instigar  
mulheres  
entrarem  
na  
tecnológica.  
Queremos  
mudar  
essa  
realidade  
garotas  
tão  
rica  
fantásticas

Shell

rica  
fantásticas  
como  
computação.  
olhe  
temos  
muita  
história  
nesse  
campo,  
viu?!  
Nas  
percebemos  
havam  
motivos  
sermos  
salas  
cursos  
universitários  
técnicos  
informática  
tecnologia  
Brasil,  
resolvemos  
organizar  
dar  
jeitinho  
nisso.  
Foi  
encontramos  
outras  
mundo,  
reuniam  
nome

```
Shell
reuniam
nome
da
programa
falavam
abstração
jeito
sabem
falar.
mostrar
s
h;
limites
sua
capacidade
intelectual.
delas
escrever
c' digo.
Façam
isso.
desenvolver
super
sistemas
embarcados.
Desenvolvam.
querem
brincar
fazendo
algumas
script
python.

python.
Brinquem.
revolucionar
mundo
mesmo
sem
ter
seu
c' digo
fonte.
Revolucionem.
Por
voc's
são
capazes.
segregar,
contrário,
queremos
criar
ambiente
sejam
atraídas
pela
tecnologia.
fazer
elas
vejam
pensem
"Eu
quero
estar
a-
também!".
```

### #13.b.

Conte quantas vezes apareceu a palavra ladies no conteúdo da página

```
import requests
from bs4 import BeautifulSoup
import re
print('Nesta atividade vamos utilizar o requests para obter o
conteúdo da seguinte página da Web:
http://brasil.pyladies.com/about')
print('E depois vamos exibir quantas vezes a palavra Ladies
apareceu no conteúdo da página seja em maiusculo ou minusculo')

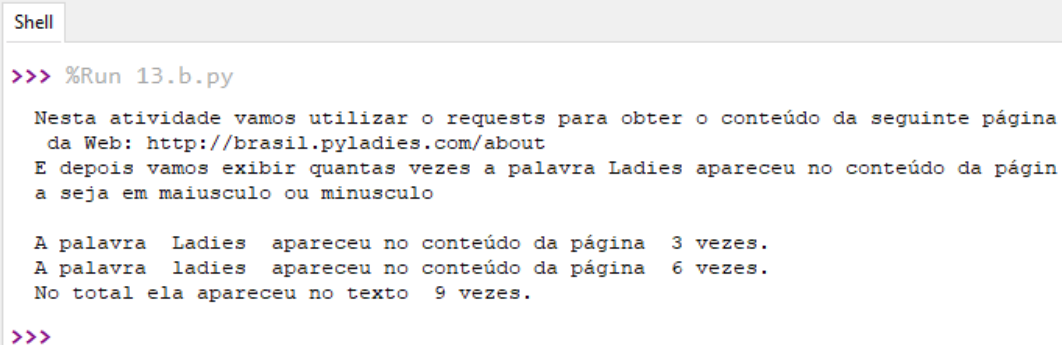
url = "http://brasil.pyladies.com/about/"

minusculta = "ladies"
maiuscula = "Ladies"

html = requests.get(url).text

soup = BeautifulSoup(html, "lxml")
M = len(re.findall(maiuscula, soup.get_text()))
m = len(re.findall(minusculta, soup.get_text()))

print("\nA palavra ", maiuscula, " apareceu no conteúdo da
página " ,M, "vezes.")
print("A palavra ", minusculta, " apareceu no conteúdo da página
" ,m, "vezes.")
print("No total ela apareceu no texto " ,M+m, "vezes.")
```



```
Shell
>>> %Run 13.b.py

Nesta atividade vamos utilizar o requests para obter o conteúdo da seguinte página
da Web: http://brasil.pyladies.com/about
E depois vamos exibir quantas vezes a palavra Ladies apareceu no conteúdo da página
a seja em maiusculo ou minusculo

A palavra Ladies apareceu no conteúdo da página 3 vezes.
A palavra ladies apareceu no conteúdo da página 6 vezes.
No total ela apareceu no texto 9 vezes.

>>>
```