****

**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](#_Toc532432831)

[**#02.** 3](#_Toc532432832)

[**#03.** 4](#_Toc532432833)

[**#04.** 4](#_Toc532432834)

[**#05.** 5](#_Toc532432835)

[**#06.** 7](#_Toc532432836)

[**#07.** 7](#_Toc532432837)

[**#08.** 8](#_Toc532432838)

[**#09.** 10](#_Toc532432839)

[**#10.** 13](#_Toc532432840)

[**#10.a.** 13](#_Toc532432841)

[**#10.b.** 15](#_Toc532432842)

[**#11.** 17](#_Toc532432843)

[**#11.a.** 17](#_Toc532432844)

[**#11.b.** 19](#_Toc532432845)

[**#11.c.** 19](#_Toc532432846)

[**#11.d.** 19](#_Toc532432847)

[**#12.** 19](#_Toc532432848)

[**#12.a.** 20](#_Toc532432849)

[**#12.b.** 20](#_Toc532432850)

[**#13.** 21](#_Toc532432851)

[**#13.a.** 22](#_Toc532432852)

[**#13.b.** 22](#_Toc532432853)

**#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 núnero 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 núnero inteiro")

else:

break

T\_crescente=(sorted(T))

print('\nA tupla criada na ordem crescente ficou da seguinte forma: ', T\_crescente)

**#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)

**#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))

**#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 apresenta-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 núnero 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 núnero inteiro")

else:

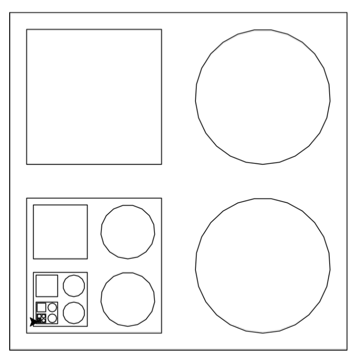
break

L\_reversa=L[::-1]

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

**#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

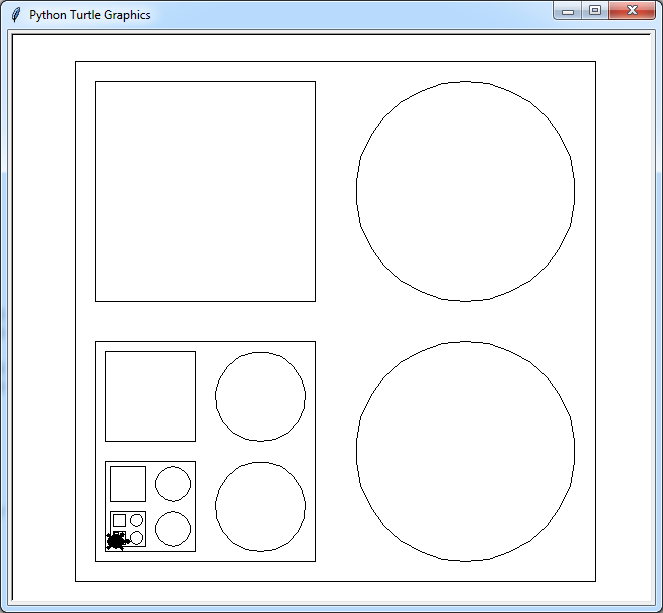
quadrado(X, Y)

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 indice 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)

**#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('Courrier ', 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('Courrier 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**

**#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

import pprint

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.')

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.')

**#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/Winter\_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()

**#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):**

**#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')

**#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 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)

**#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')

**#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=''

**#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/"

minuscula = "ladies"

maiuscula = "Ladies"

html = requests.get(url).text

soup = BeautifulSoup(html, "lxml")

M = len(re.findall(maiuscula, soup.get\_text()))

m = len(re.findall(minuscula, soup.get\_text()))

print("\nA palavra ", maiuscula, " apareceu no conteúdo da página " ,M, "vezes.")

print("A palavra ", minuscula, " apareceu no conteúdo da página " ,m, "vezes.")

print("No total ela apareceu no texto " ,M+m, "vezes.")