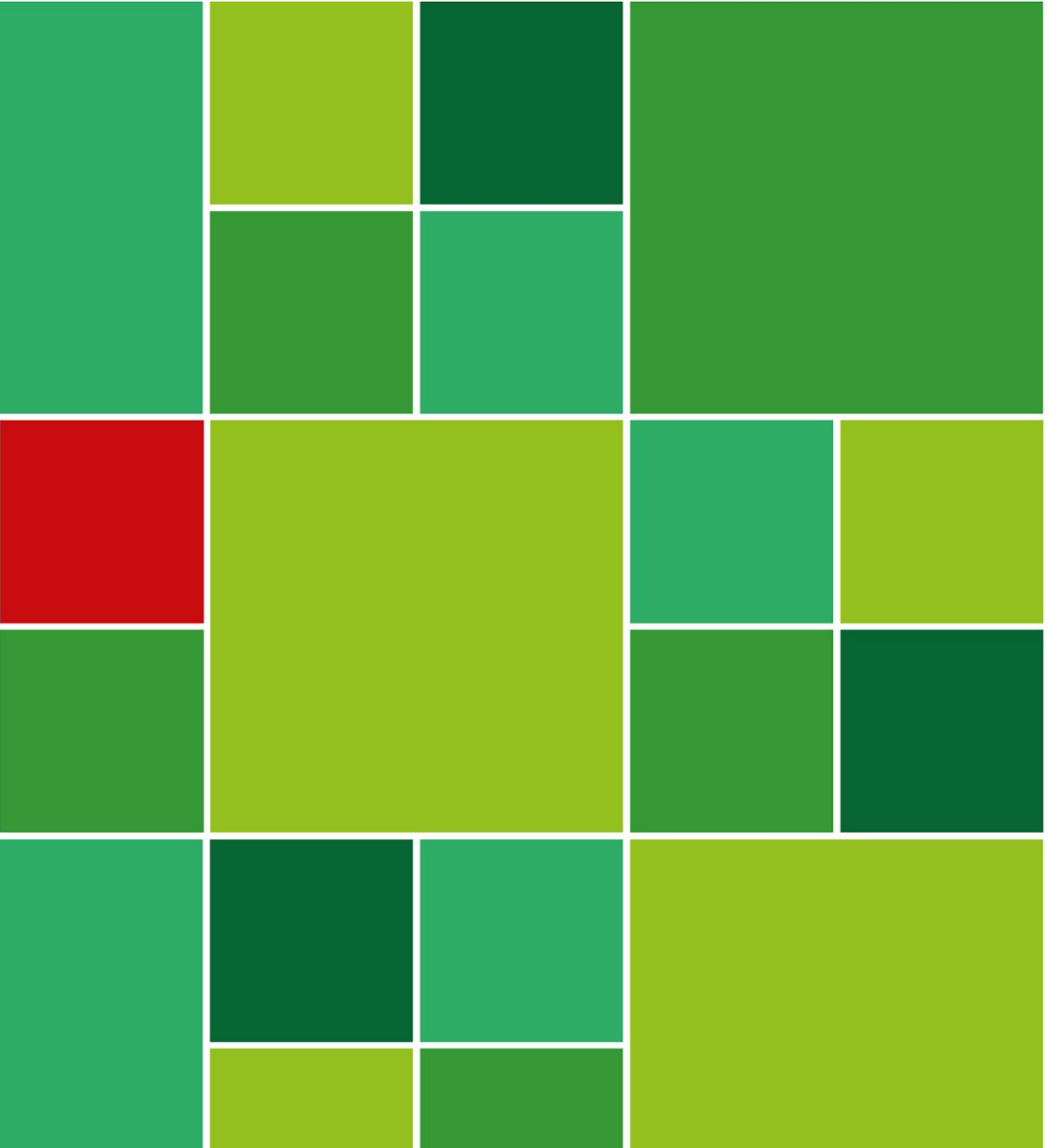


Estrutura de Dados

Listas e Tuplas

Prof. Dr. Danilo Barbosa

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PERNAMBUCO

Conteúdo

- Conceito de Listas
- Conceito de tuplas
- Concatenação de tuplas
- Empacotamento e despacotamento de tuplas
- Atividade
- Referências

Conceito de Listas

A lista é uma estrutura de dados que agrupa objetos sequencialmente.

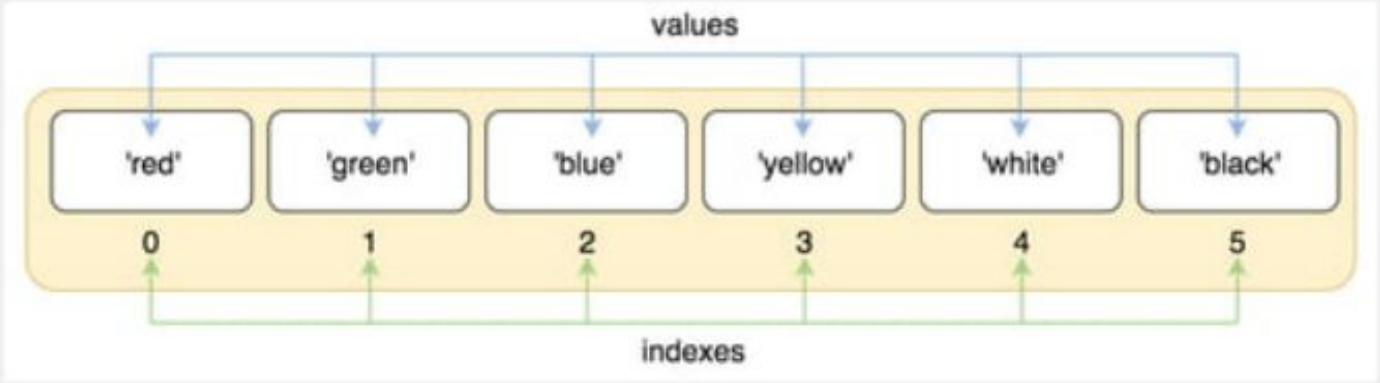


Figura: Ilustração de uma estrutura de lista com 6 objetos do tipo **str** (texto)

- Cada item da lista é identificado por um **índice**.
- O primeiro item da lista tem índice o

Criação de Listas

- Uma lista é definida em Python como uma sequência de valores ou objetos entre colchetes.
- A lista pode ser atribuída a uma variável, e então todos os objetos da lista serão acessíveis por meio dessa única variável.
- É possível mostrar o conteúdo da lista com **print**.

Exemplo de código

```
lista = [10, 20, 30, 40] #lista com 4 objetos int
print(lista) #mostra a lista no terminal
```

Criação de Listas

Exemplo de código – lista de **int**

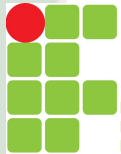
```
lista = [10, 20, 30, 40] #lista com 4 objetos int
print(lista) #mostra a lista no terminal
```

Exemplo de código – lista de **str**

```
#lista com 3 objetos str
lista = ['Filipe', 'Joao', 'Ana']
print(lista) #mostra a lista no terminal
```

Exemplo de código – lista de **str** e **int**

```
#lista com 6 objetos str e int
lista = ['Filipe', 10, 'Joao', 20, 'Ana', 30]
print(lista) #mostra a lista no terminal
```



Índice de acesso de Listas

Para acessar um item da lista, você deve digitar o nome da variável seguido pelo índice do item entre colchetes, com a seguinte sintaxe: <variável>[<índice>].

Exemplo de código

```
lista = ['Filipe', 'Joao', 'Ana']
print(lista[0]) #escreve Filipe
print(lista[1]) #escreve Joao
print(lista[2]) #escreve Ana
```

Lembre-se sempre de que o primeiro índice é zero!

Atribuições de Listas

É possível modificar o valor ou objeto condido em uma posição da lista fazendo uma atribuição com a seguinte sintaxe: <variável>[<índice>]=<valor>.

Exemplo de código

```
lista = ['Filipe', 'Joao', 'Ana']
print(lista) #mostra a lista
lista[1] = 'Matheus' #substitui Joao por Matheus
print(lista) #mostra a lista modificada
```

Iterações de Listas

É possível fazer uma iteração com **for** para acessar cada objeto da lista de duas formas diferentes:

- **Utilizando índices** – um **for** é utilizado para iterar pelos índices da lista, e os acessos ou modificações são feitos com índices.
- **Sem utilizar índices** – um **for** é utilizado para iterar pelos itens da lista diretamente, sem usar índices.

Iterações de Listas

Utilize a função **len** para obter o tamanho da lista.

Iteração com índice

```
lista = ['Filipe', 'Joao', 'Ana]
for i in range(len(lista)):
    print(lista[i]) #mostra cada item da lista
```

Iteração sem índice

```
lista = ['Filipe', 'Joao', 'Ana]
for item in lista:
    print(item) #mostra cada item da lista
```

Inserção de elementos da Listas

É possível inserir objetos na lista em tempo de execução utilizando o método **append**, com a seguinte sintaxe:
<variável>.append(<valor>)

Exemplo de código

```
lista = ['Filipe', 'Joao', 'Ana']
print(lista) #Mostra a lista

#Insere 'Matheus' no final da lista
lista.append('Matheus')

print(lista) #Mostra a lista modificada
```

Remoção de elementos da Listas

É possível remover objetos da lista em tempo de execução utilizando o operador **del**, com a seguinte sintaxe: `del <variável>[<índice>]`

Exemplo de código

```
lista = ['Filipe', 'Joao', 'Ana']
print(lista) #Mostra a lista
del lista[1] #Remove 'Joao' da lista
print(lista) #Mostra a lista modificada
```

Fatias da Listas

Exemplo de código – *slicing*

```
lista = [10, 24, 3, 5, 8, 29, 11] #cria lista

fatia = lista[1:5] #obtem fatia do 24 ate o 8

print(fatia) #mostra a fatia
```

Reversão de Listas

É possível obter uma lista “de trás para frente” com a seguinte sintaxe: <variável>[::-1]

Exemplo de código

```
lista = [10, 24, 3, 5, 8, 29, 11] #cria lista

lista_reversa = lista[::-1]

#mostra lista de tras para frente
print(lista_reversa)
```

Ordenação de Listas

É possível ordenar uma lista de números com o método **sort**.

- Em ordem crescente: `<variável>.sort()`
- Em ordem decrescente: `<variável>.sort(reverse=True)`

Exemplo de código

```
lista = [10, 24, 3, 5, 8, 29, 11] #cria lista
lista.sort() #ordena lista
print(lista) #mostra lista ordenada
```

Conceito de tuplas

- Tuplas podem ser vistas como **listas em Python**. No entanto, elas **não podem ser modificadas**.

- Tuplas são ideais para representar **listas de valores constantes** e para realizar **operações de empacotamento e desempacotamento de valores**.

Conceito de tuplas

As tuplas podem ser criadas de maneira semelhantes a listas

- tupla = ()
- tupla = tuple()
- tupla = (1,2,3)
- lista = [1,2,3]
- tupla = tuple(lista)
- tupla = 1, 2, 3

Conceito de tuplas

- Tuplas suportam a maior parte das operações de listas, tais como fatiamento, indexação, tamanho
- tupla[0]
- tupla[1:]
- tupla[:]
- len(tupla)

```
for e in tupla:
    print(e)
for i, e in enumerate(tupla):
    print(f"e[{i}]={e}")
```

Concatenação de tuplas

Embora não possamos alterar uma tupla, podemos concatená-la

- t1 = (1,2,3)
- t2 = 4, 5, 6
- t1+t2
- (1,2,3,4,5,6)

Concatenação de tuplas

Mas essa tupla contiver uma lista ou outro objeto que possa ser alterado, esse objeto pode ser alterado

- t3 = (1, [2, 3, 4])
- len(t3)
- t3[1]
- t3[1].append(8)
- t3

empacotamento e despacotamento de tuplas

Empacotamento / Desempacotamento

- a = 1, 2
- c, d = (3, 4)

- *a, b = [1,2,3,4,5]
- a,*b = [1,2,3,4,5]

Atividade

- Cada equipe deve fazer dois códigos de tuplas e listas da vídeo aula. Em seguida, enviar o print das atividades para o google sala de aula .
- Quatro equipes serão sorteadas para explicar o código durante a aula

Atividade

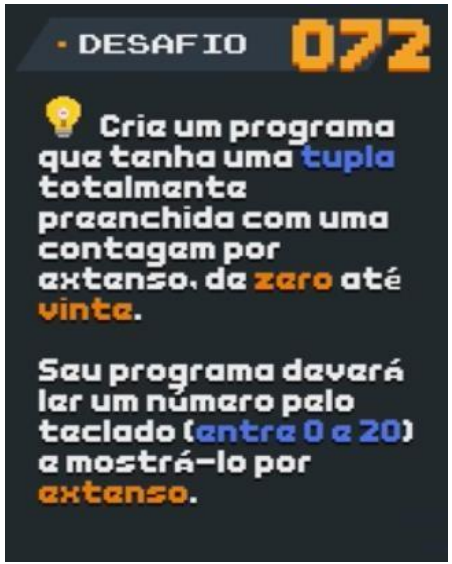
- Equipe 1 - questões 76 e 77 Igor / Ian
- Equipe 2- questões 78 e 79 Pedro
- Equipe 3 - questões 80 e 81 Vinicius / Luiz Felipe
- Equipe 4 - questões 82 e 83 Isabela / Leandro
- Equipe 5 - questões 84 e 85 Débora / Isabelly

Atividade

- Equipe 6 - questões 86 e 87 Brenda / Lane / Alana
- Equipe 7 - questões 88 e 89 Paulo
- Equipe 8 - questões 90 e 91 Luiz/ Ronaldo / Gabriel
- Equipe 9 - questões 92 e 93
- Equipe 10 - questões 94 e 95

Atividade

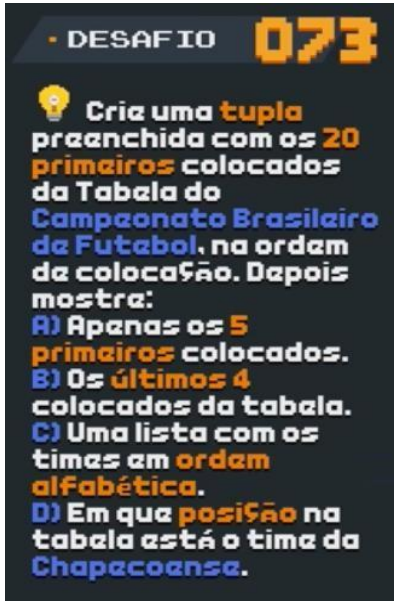
- Desenvolva esses códigos em Python:




```
cont = ('zero', 'um', 'dois', 'três', 'quatro',
        'cinco', 'seis', 'sete', 'oito', 'nove',
        'dez', 'onze', 'doze', 'treze', 'catorze',
        'quinze', 'dezesseis', 'dezessete', 'dezoito',
        'dezenove', 'vinte')
while True:
    núm = int(input('Digite um número entre 0 e 20: '))
    if 0 <= núm <= 20:
        break
    print('Tente novamente. ', end='')
print(f'Você digitou o número {cont[]}')
```

Atividade

- Desenvolva esses códigos em Python:

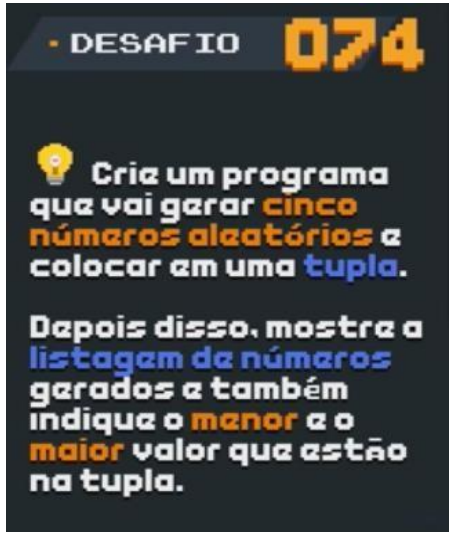



```
times = ('Corinthians', 'Palmeiras', 'Santos', 'Grêmio',
         'Cruzeiro', 'Flamengo', 'Vasco', 'Chapecoense',
         'Atlético', 'Botafogo', 'Atlético-PR', 'Bahia',
         'São Paulo', 'Fluminense', 'Sport Recife',
         'EC Vitória', 'Coritiba', 'Avaí', 'Ponte Preta',
         'Atlético-GO')
print('-=' * 15)
print(f'Lista de times do Brasileirão: {times}')
print('-=' * 15)
print(f'Os 5 primeiros são {times[0:5]}')
print('-=' * 15)
print(f'Os 4 últimos são {times[-4:]}')
print('-=' * 15)
print(f'Times em ordem alfabética: {sorted(times)}')
```

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

Atividade

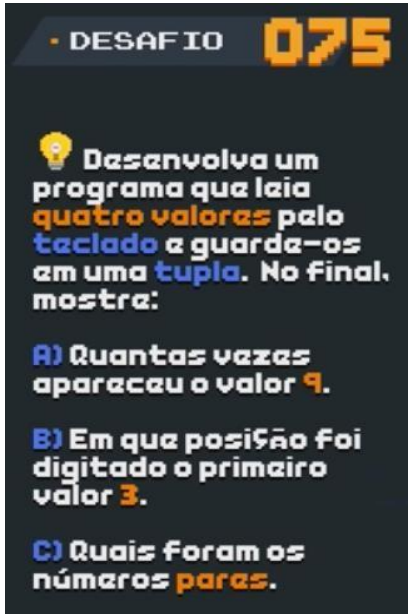
- Desenvolva esses códigos em Python:

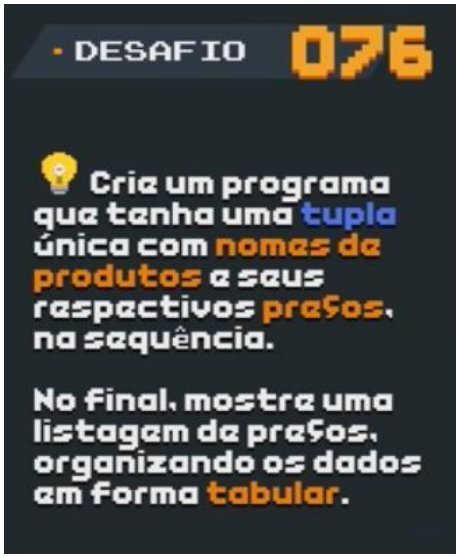


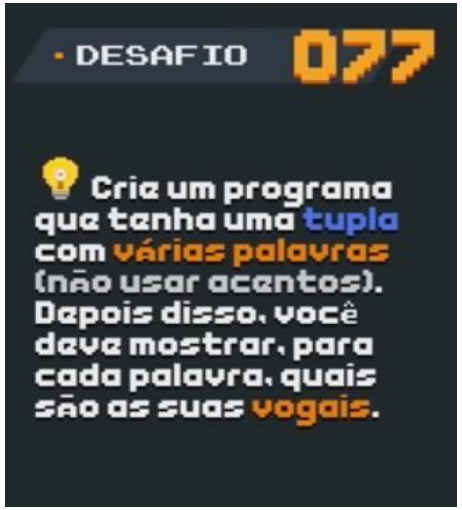

```python
from random import randint
numeros = (randint(1, 10), randint(1, 10), randint(1, 10),
     randint(1, 10), randint(1, 10))
print('Os valores sorteados foram: ', end='')
for n in numeros:
    print(f'{n} ', end='')
print(f'\nO maior valor sorteado foi {max(numeros)}')
print(f'O menor valor sorteado foi {min(numeros)}')
```

Atividade

- Desenvolva esses códigos em Python:



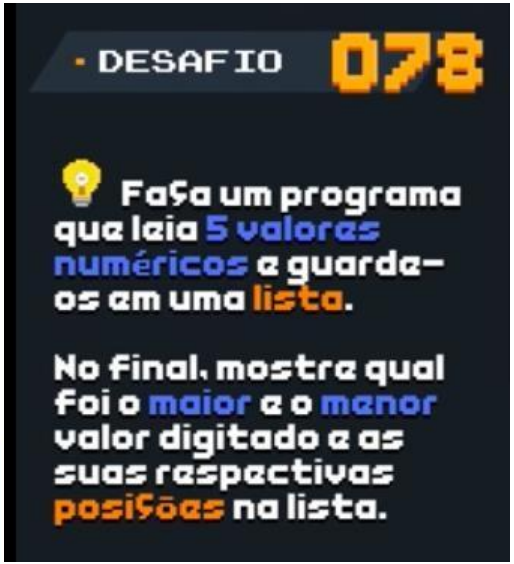




INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

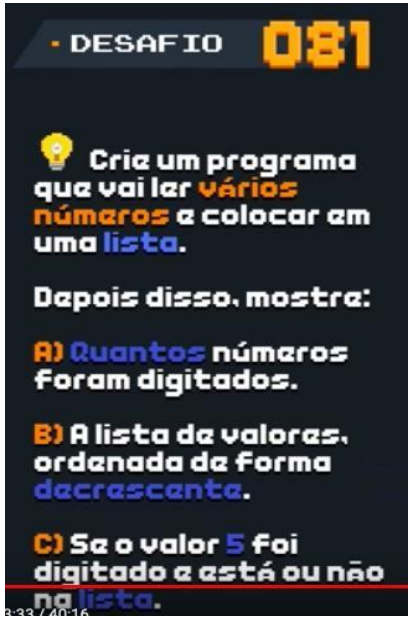
Atividade

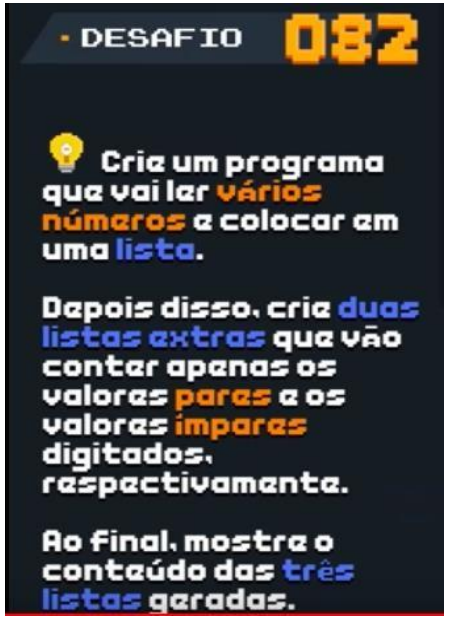
- Desenvolva esses códigos em Python:



Atividade

- Desenvolva esses códigos em Python:

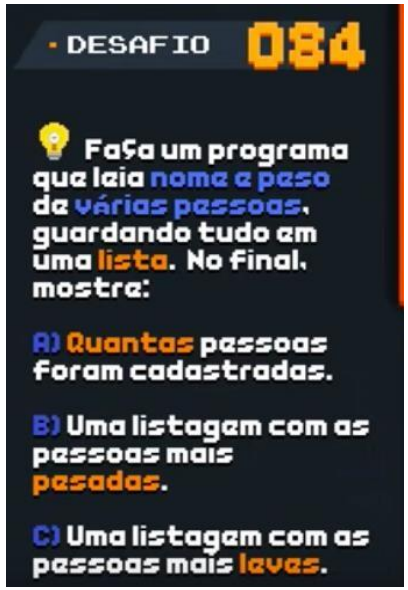


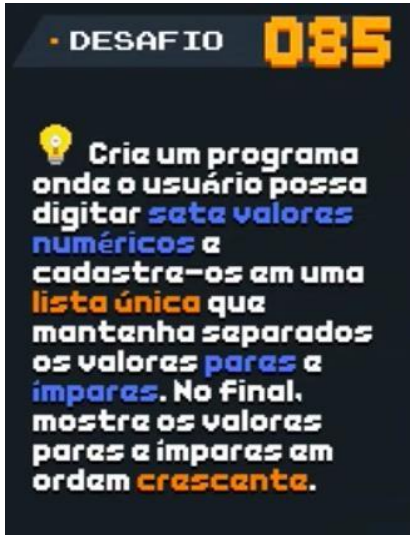


INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

Atividade

- Desenvolva esses códigos em Python:

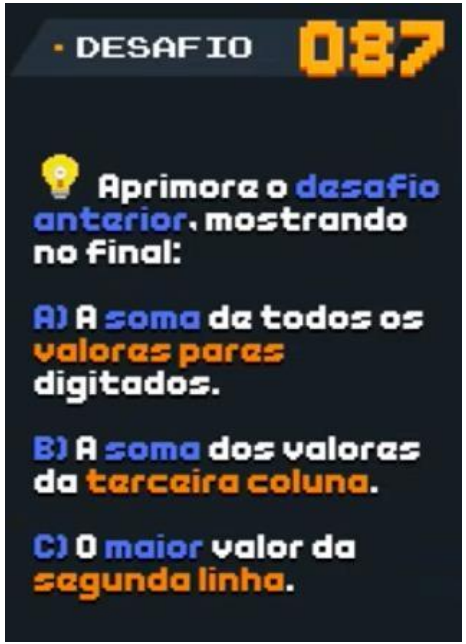


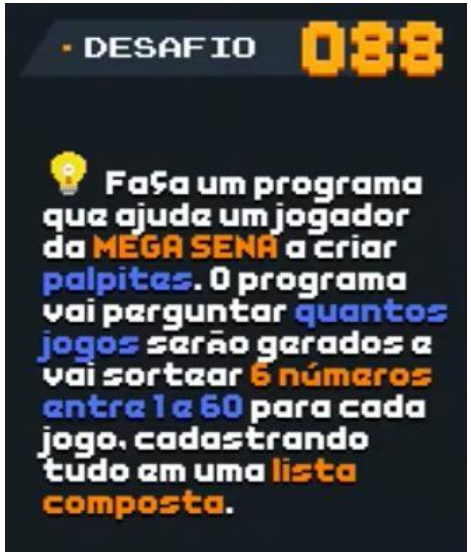


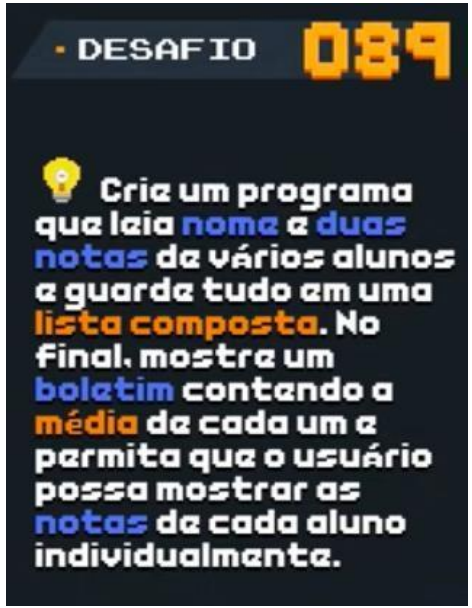
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

Atividade

- Desenvolva esses códigos em Python:



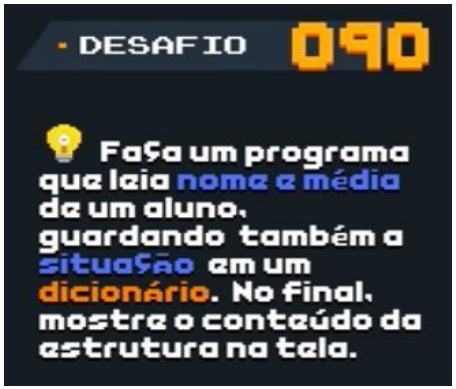


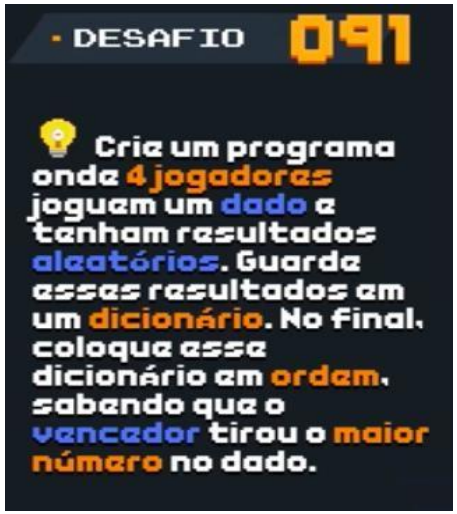


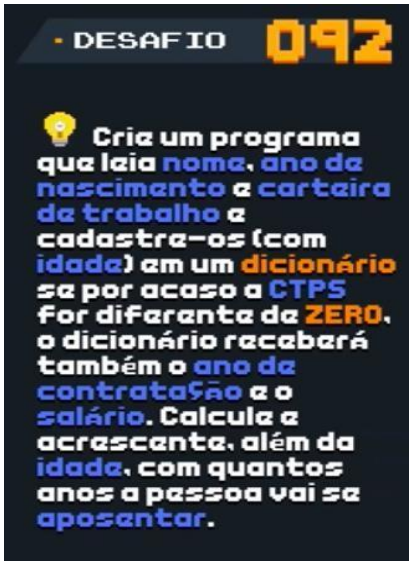
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

Atividade

- Desenvolva esses códigos em Python:



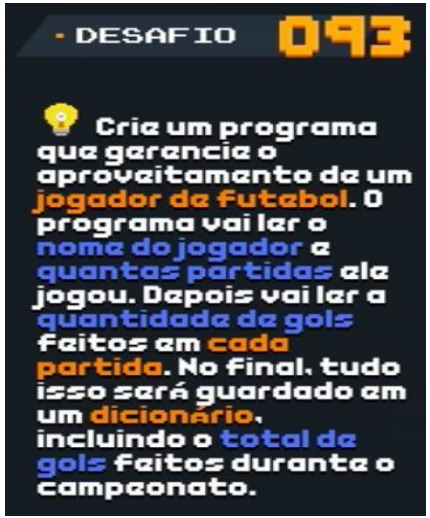


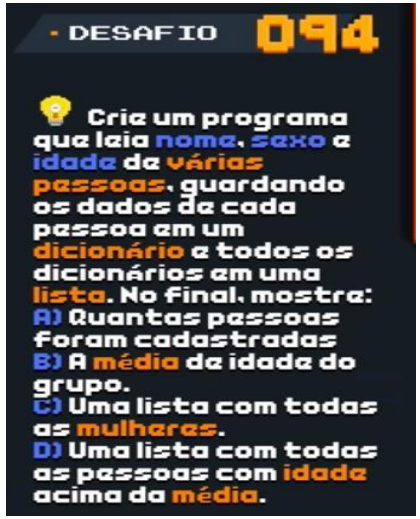


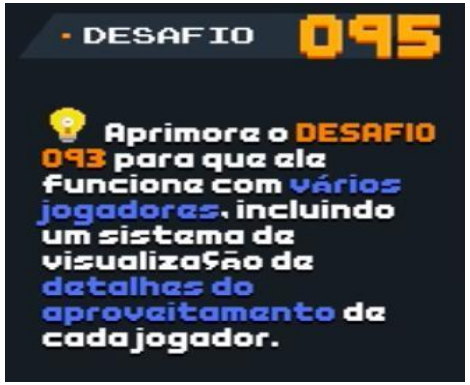
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

Atividade

- Desenvolva esses códigos em Python:







INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PERNAMBUCO

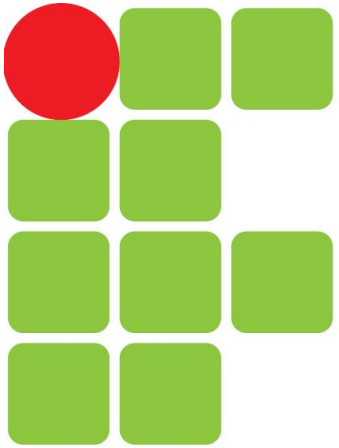
Referências

PUGA, Sandra; RISSETTI, Gerson. Lógica de Programação e Estruturas de Dados-Com Aplicações em Java. 3ª edição. 2016.

GOODRICH, Michael T.; TAMASSIA, Roberto. **Estruturas de dados & algoritmos em Java**. Bookman Editora, 2013.

CASTILHO, Marcos; SILVA, Fabiano; WEINGAERTNER, Daniel. Algoritmos e Estruturas de Dados I. 2011.

Curso de programação em Python do prof. Gustavo Guanabara. Último acesso em: 01 Set 2024.



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PERNAMBUCO