Mundo 3 - Dicionários

1) Faça um programa que leia nome e média de um aluno, guardando também a situação em um dicionário. No final, mostre o conteúdo da estrutura na tela.

a)

```
aluno = dict()
aluno['Nome'] = input('Nome: ')
aluno['Media'] = float(input('Média: '))
aluno['Situação'] = ''
for v in aluno.items():
    if aluno['Media'] >= 7:
        aluno['Situação'] = 'Aprovado'
    else:
        aluno['Situação'] = 'Reprovado'
print('-='*30)
for k,v in aluno.items():
    print(f'- {k} é igual a {v}')
```

Guanabara)

A única diferença é a firula no input do aluno[’media’] e a forma como o condicionar é determinado, onde se

```
aluno = dict()
aluno['nome'] = input('Nome: ')
aluno['média'] = float(input(f'Média de {aluno["nome"]}: '))
if aluno['média'] >= 7:
    aluno['situação'] = 'Aprovado'
elif 5 <= aluno['média'] < 7:
    aluno['situação'] = 'Recuperação'
else:
    aluno['situação'] = 'Reprovado'
print()
print('-='*30)
for k,v in aluno.items():
    print(f'{k} é igual {v}')
```

2) Crie um programa onde 4 jogadores joguem um dado e tenham resultados aleatórios. Guarde esses resultados em um dicionário. No final, coloque esse dicionário em ordem, sabendo que o vencedor tirou o maior número do dado.

a)

Tive a mesma solução, a única diferença foi a forma como eu criei o dicionário.

```
from random import randint
from time import sleep
from operator import itemgetter
jogadores = dict()
jogadores['jogador1'] = randint(1,6)
jogadores['jogador2'] = randint(1,6)
jogadores['jogador3'] = randint(1,6)
jogadores['jogador4'] = randint(1,6)
print('Valores sorteados:')
for k,v in jogadores.items():
    print(f'{k} tirou {v} no dado. ')
    sleep(0.5)
print('-='*30)
print('  ==RANKING DOS JOGADORES==')
ranking = sorted(jogadores.items(), key=itemgetter(1), reverse=True)
for i, v in enumerate(ranking):
    print(f'  {i+1}º lugar: {v[0]} com {v[1]}')
    sleep(0.5)
```

Guanabara)

Este código apresenta funções novas que não foram apresentadas, então vou mostrar passo por passo:

- Importação de randint, time e uma nova biblioteca chamada operator, com ela temos várias opções de calculo e puxar informações de outras estruturas.
- Criamos um dicionários chamando jogadores onde temos várias keys de jogaroesx recebendo um randint entre 1 a 6
- printamos na tela “Valores sorteados”
- Agora vamos varrer as informações do dicionário mostrando que o jogador tal tirou o randint, dando um sleep de 0.5 em cada loop
- Feito isso damos um print de informações na tela
- Criamos uma segunda lista que será mostrado a ordem dos resultados das classificações, onde ranking recebe o sorted de jogadores.items() e a key para organizar será o método itemgetter na posição 1 que é o valor random, o reverse true para mostrar do maior para o menor
- Fazemos a última varredura da lista ranking mostrando o lugar o valor e o jogador

```
from random import randint
from time import sleep
from operator import itemgetter
jogadores = {
    'jogador1': randint(1,6),
    'jogador2': randint(1,6),
    'jogador3': randint(1,6),
    'jogador4': randint(1,6)
}
print('Valores sorteados:')
for k,v in jogadores.items():
    print(f'{k} tirou {v} no dado. ')
    sleep(0.5)
print('-='*30)
print('  ==RANKING DOS JOGADORES==')
ranking = sorted(jogadores.items(), key=itemgetter(1), reverse=True)
for i, v in enumerate(ranking):
    print(f'  {i+1}º lugar: {v[0]} com {v[1]}')
    sleep(0.5)
```

3) Crie um programa que leia nome, ano de nascimento e carteira de trabalho e cadastre-os (com idade) em um dicionário se por acaso a CTPS for diferente de 0, o dicionário recebera também o ano de contratação e o salário. Calcule e acrescente além da idade, com quantos anos a pessoa vai se aposentar

- Mulheres = 62 anos + 15 anos de contribuição / 30 anos de contribuição + 58 anos
- Homem = 65 anos + 15 anos de contribuição / 35 anos de contribuição + 63 anos

a)Não consegui seguir com a minha ideia, vou parar para estudar IMSO

```
# Dados iniciais
cadastro={}
def dados():
    import datetime
    global cadastro
    cadastro = {
        'Nome': input('Nome: '),
        'Ano nascimento': int(input('Ano Nascimento: ')),
        'CTPS': int(input('CTPS ( 0 se não tiver ): ')),
        'SEXO': input('SEXO [M/F]: '.strip().upper())
    }
    cadastro['Idade'] = datetime.datetime.today().year - cadastro['Ano nasc
imento']
    if cadastro['CTPS'] != 0:
        cadastro['Ano Contratação'] = int(input('Ano de Contratação: '))
```

```
        cadastro['Salário'] = float(input('Salário: '))
        cadastro['Tempo Contribuição'] = datetime.datetime.today().year - c
adastro['Ano Contratação']
    return cadastro

#Verificadores
def por_idade(cadastro):
    if cadastro['SEXO'] in 'Mm':
        if cadastro['Idade'] >= 65:
            if cadastro['Tempo Contribuição'] >= 15:
                cadastro['Aposentadoria'] = 'Aprovado por Idade'
            else:
                cadastro['Aposentadoria'] = 'Reprovado por Idade'
    elif cadastro['SEXO'] in 'Ff':
        if cadastro['Idade'] >= 62:
            if cadastro['Tempo Contribuição'] >= 15:
                cadastro['Aposentadoria'] = 'Aprovado por Idade'
            else:
                cadastro['Aposentadoria'] = 'Reprovado por Idade'
    return cadastro
def por_tempo(cadastro):
    if cadastro['SEXO'] in 'Mm':
        if cadastro['Idade'] >= 63:
            if cadastro['Tempo Contribuição'] >= 35:
                cadastro['Aposentadoria'] = 'Aprovado por tempo de contribu
ição'
            else:
                cadastro['Aposentadoria'] = 'Reprovado por tempo de contrib
uição'
    elif cadastro['SEXO'] in 'Ff':
        if cadastro['Idade'] >= 58:
            if cadastro['Tempo Contribuição'] >= 30:
                cadastro['Aposentadoria'] = 'Aprovado por por tempo de cont
ribuição'
            else:
                cadastro['Aposentadoria'] = 'Reprovado por por tempo de con
tribuição'
    return cadastro
def resultado(cadastro):
    print('  == Resultado ==')
    for k,v in cadastro.items():
        print(f'  {k}: {v}')

#Menu
def menu():
    global cadastro
    while True:
        print('[ 1 ] Por tempo / [ 2 ] Por idade')
```

```
        esc = int(input('Escolha uma das opções: '))
        if esc == 1:
            cadastro = por_tempo(cadastro)
            resultado(cadastro)
            break
        elif esc == 2:
            cadastro = por_idade(cadastro)
            resultado(cadastro)
            break
        else:
            esc = int(input('Opção inválida, tente novamente: '))

cadastro = dados()
menu()
```

b) Este exercício na verdade fiz errado, pois era para verificar quanto tempo falta para aposentar, e não se eu tenho direito ou não, mas é uma boa solução para verificar.

```
import datetime
cadastro = {
    'Nome': input('Nome: '),
    'Ano nascimento': int(input('Ano Nascimento: ')),
    'CTPS': int(input('CTPS ( 0 se não tiver ): ')),
    'SEXO': input('SEXO [M/F]: '.strip().upper())
}
cadastro['Idade'] = datetime.datetime.today().year - cadastro['Ano nascimen
to']
if cadastro['CTPS'] != 0:
    cadastro['Ano Contratação'] = int(input('Ano de Contratação: '))
    cadastro['Salário'] = float(input('Salário: '))
    cadastro['Tempo Contribuição'] = datetime.datetime.today().year - cadas
tro['Ano Contratação']
print('-='*30)
if cadastro['SEXO'] in 'Mm' and cadastro['Idade'] >= 65 and cadastro['Tempo
Contribuição'] >= 15:
    cadastro['Aposentadoria'] = 'Aprovado por Idade'
elif cadastro['SEXO'] in 'Mm' and cadastro['Idade'] >= 63 and cadastro['Tem
po Contribuição'] >= 35:
    cadastro['Aposentadoria'] = 'Aprovado por Tempo de Contribuição'
```

```
elif cadastro['SEXO'] in 'Ff' and cadastro['Idade'] >= 62 and cadastro['Tem
po Contribuição'] >= 15:
    cadastro['Aposentadoria'] = 'Aprovado por Idade'
elif cadastro['SEXO'] in 'Ff' and cadastro['Idade'] >= 58 and cadastro['Tem
po Contribuição'] >= 30:
    cadastro['Aposentadoria'] = 'Aprovado por Tempo de Contribuição'
else:
    cadastro['Aposentaria'] = 'Reprovado, não possui tempo e / ou Idade'
print('  ==Dados Pessoais ==')
for k,v in cadastro.items():
    print(f'  {k}: {v}')
```

Guanabara)

O Guanabara apenas adicionou 35 anos ao ano de contratação para saber se foi aposentado ou não.

```
import datetime
cadastro = {
    'Nome': input('Nome: '),
    'Ano nascimento': int(input('Ano Nascimento: ')),
    'CTPS': int(input('CTPS ( 0 se não tiver ): ')),
    'SEXO': input('SEXO [M/F]: '.strip().upper())
}
cadastro['Idade'] = datetime.datetime.today().year - cadastro['Ano nascimen
to']
if cadastro['CTPS'] != 0:
    cadastro['Ano Contratação'] = int(input('Ano de Contratação: '))
    cadastro['Salário'] = float(input('Salário: '))
    cadastro['Idade para aposentar'] = cadastro['Idade'] + ((cadastro['Ano
Contratação']+35) - datetime.datetime.now().year)
print('-='*30)
print('  ==Dados Pessoais ==')
for k,v in cadastro.items():
    print(f'  {k} =  {v}')
```

4) Crie um programa que gerencie o aproveitamento de um jogador de futebol. O programa vai ler o nome do jogador e quantas partidas ele jogou. Depois vai ler a quantidade de gols feitos em cada partida. No final, tudo isso será guardado em um dicionário. Incluindo o total de gols feitos durante o campeonato

a) Este exercício podemos fazer de algumas formas diferentes, a primeira foi criar o dicionário e ir inserindo as keys e values nele, após isso é só questão de varrer dados e formatando resultados.

```
jogador = dict()
jogador['Nome'] = input('Nome: ')
partidas = int(input('Quantidas partirdas jogou?: '))
```

```
jogador['Gols'] = []
for c in range(partidas):
    jogador['Gols'].append(int(input(f'Quantidade de gols na partida {c}:
')))
total_gols = sum(jogador['Gols'])
jogador['Total de Gols'] = total_gols
print('-='*32)
print(jogador)
print('-='*32)
for k, v in jogador.items():
    print(f'O campo {k} tem valor de: {v}')
print('-='*32)
print(f'O jogador {jogador["Nome"]} jogou {len(jogador["Gols"])}')
for i, c in enumerate(jogador['Gols']):
    print(f'=> Na partida {i}, fez {c} gols.')
print(f'No total foram {jogador["Total de Gols"]}')
```

b) Podemos criar o dicionário já com os campos a serem utilizados e manipular a entrada e saída desses resultados

```
jogador = {
    'Nome': input('Nome: '),
    'Partidas': int(input('Quantas partidas jogou?: ')),
    'Gols': [],
    'Total de Gols':int()
}
for c in range(jogador['Partidas']):
    jogador['Gols'].append(int(input(f'Quantidade de gols na partida {c}:
')))
jogador['Total de Gols'] = sum(jogador['Gols'])
print('-='*32)
print(jogador)
print('-='*32)
for k, v in jogador.items():
    print(f'O campo {k} tem valor de: {v}')
print('-='*32)
print(f'O jogador {jogador["Nome"]} jogou {len(jogador["Gols"])}')
for i, c in enumerate(jogador['Gols']):
    print(f'=> Na partida {i}, fez {c} gols.')
print(f'No total foram {jogador["Total de Gols"]}')
```

Guanabara)
O professor fez de uma forma diferente, onde o dicionário não tinha nenhum valor e ele foi criando a cada passo e inserindo individualmente.

```
jogador = dict()
partidas = list()
jogador['Nome'] = input('Nome do jogador: ')
tot = int(input(f'Quantas partidas {jogador["Nome"]} jogou?: '))
for c in range(0,tot):
    partidas.append(int(input(f'  Quantos gols na partida {c}: ')))
jogador['gols'] = partidas[:]
jogador['total'] = sum(partidas)
print('-='*30)
print(jogador)
print('-='*30)
for k,v in jogador.items():
    print(f'O campo {k} tem o valor {v}')
print('-='*30)
print(f'O jogador {jogador["Nome"]} jogou {len(jogador["gols"])} partida
s.')
for i, v in enumerate(jogador['gols']):
    print(f'   => Na partida {i}, fez {v} gols')
print(f'Foi um total de {jogador["total"]} gols.')
```

5) Crie um programa que leia nome, sexo e idade de várias pessoas, guardando os dados de cada pessoa em um dicionário e todos os dicionários em uma lista. No final mostre:

- Quantas pessoas forma cadastradas
- A média de idade do grupo
- Uma lista com todas as mulheres
- Uma lista com todas as pessoas com idade acima da média

a) Esta foi a minha solução, vou estar olhando agora a solução do professor Guanabara

```
pessoas = list()
mulheres = list()
tot_idade = media = 0
while True:
    pessoas.append(dict(Nome = input('Nome: '))) # INSERIR NOME NO DICIONÁR
IO
    sexo = input('Sexo [M/F]: ')

    while sexo not in 'MmFf' or sexo =='':
        print('ERRO ! Por favor, digite apenas M ou F.') # VALIDAÇÃO DE M /
F CORRETO
        sexo = input('Sexo [M/F]: ')
    if sexo in 'Ff':
```

```
        mulheres.append(pessoas[len(pessoas)-1]['Nome']) # ACRESCENTA NOME
DAS MULHERES

    pessoas[len(pessoas)-1]['Sexo'] = sexo
    pessoas[len(pessoas)-1]['Idade'] = int(input('Idade: ')) # ACRESCENTA I
DADE NO DICIONÁRIO
    tot_idade += pessoas[len(pessoas)-1]['Idade']

    resp = input('Quer continuar?: ')
    while resp not in 'SsNn':
        print('ERRO! Responda apenas S ou N') # VALIDAÇÃO DE S OU N CORRETA
        resp = input('Quer continuar?: ')
    if resp in 'Nn' and resp != '':
        break

media = tot_idade / len(pessoas) # CALCULO DE MÉDIA


print('-='*30)
print(f'A ) Ao todo temos {len(pessoas)} cadastradas.')
print(f'B ) A média de idade é de {media}')
print(f'C ) As mulheres cadastradas foram: ',end='')
for c in mulheres:
    print(c, end=', ')
print()
print(f'D ) Lista das pessoas que estão acima da média')
for i, p in enumerate(pessoas):
    if pessoas[i]['Idade'] > media:
        for k,v in p.items():
            print(f'{k:<5} = {v:<8}',end=';')
        print()
print('<< ENCERRADO >>')
```

Guanabara)

Vamos entender como está funcionando o código do professor:

- Declaramos um dicionário chamado pessoa
- Declaramos uma lista chamada galera

Enquanto verdade

- Limpamos o dicionário pessoa
- key nome em pessoas recebe um input para nome
- Enquanto verdade
    - key sexo em pessoa recebe um input, em maiúsculo e apenas a primeira letra
    - se key sexo estiver em MmFf então ele sai do primeiro laço

    - se não, ele da um erro e volta para o input
- key idade de pessoa recebe um input
- soma recebe soma mais key idade em pessoa
- lista galera vai receber um append da copia da pessoa

Enquanto verdade

- resp recebe um input em maiúscula e somente a primeira letra
- se resp estiver em SN então ele dá break do primeiro loop
- senão ele dá erro e volta pro input
- se resp for n ele dá break do loop geral

No final são representados alguns dados, sendo:

- Uma linha para firular
- O total de pessoas cadastradas sendo o len da lista, lembrando que dentro da lista existem vários dicionários.
- variável média recebe soma dividida pela len de galera
- printamos a média de idade

Abaixo vamos printar as mulheres que foram cadastradas, da seguinte forma:

- printamos a frase (as mulheres cadastradas foram:) com um end = '' então a proxima linha junta
- varremos a lista galera e se p['sexo'] for f então ele printa o nome com end = '' no final
- abaixo printamos também as pessoas com idade acima da média, fazendo da mesma forma anterior, varremos a lista e se p['idade'] for maior que idade então ele printa o nome e a idade

```
pessoa = dict()
galera = list()
soma = media = 0
while True:
        pessoa.clear()
    pessoa['Nome'] = input('Nome: ')
    while True:
        pessoa['Sexo'] = input('Sexo [M/F]: ').upper()[0]
        if pessoa['Sexo'] in 'MmFf':
            break
        print('ERRO! Por favor, digite apenas M ou F.')
    pessoa['Idade'] = int(input('Idade: '))
    soma += pessoa['Idade']
    galera.append(pessoa.copy())
    while True:
        resp = input('Quer continuar [S/N]?: ').upper()[0]
        if resp in 'SN':
            break
        print('Responda apenas S ou N')
```

```
    if resp == 'N':
        break
print('-='*30)
print(f'A ) Ao todo temos {len(galera)} pessoas cadastradas')
media = soma / len(galera)
print(f'B ) A média de idade é de {media:5.2f} anos.')
print('C ) As mulheres cadastradas foram ',end='')
for p in galera:
    if p['Sexo'] in 'Ff':
        print(f'{p['Nome']} ', end='')
print()
print('D ) Lista das pessoas que estão acima da média: ')
for p in galera:
    if p['Idade'] >= media:
        print('     ',end='')
        for k, v in p.items():
            print(f'{k} = {v}; ', end='')
        print()
print('<<ENCERRADO>>')
```

6) Aprimore o desafio 4 para que ele funcione com vários jogadores, incluindo um sistema de visualização de detalhes do aproveitamento de cada jogador

a)

```
jogadores = []
jogador = dict()
while True:
    jogador.clear()
    jogador['Nome'] = input('Nome: ')
    partidas = int(input('Quantidas partirdas jogou?: '))
    jogador['Gols'] = []
    for c in range(partidas):
        jogador['Gols'].append(int(input(f'Quantidade de gols na partida
{c}: ')))
    total_gols = sum(jogador['Gols'])
    jogador['Total de Gols'] = total_gols
    jogadores.append(jogador.copy())
    while True:
        resp = input('Quer continuar [S/N]?: ').upper()
        if resp in 'SN':
            break
        print('Responda apenas S ou N')
    if resp =='N':
        break
print('-='*30)
```

```
print(f'{"cod":<3} {"nome":<10} {"gols":<20} {"total":<10}')
print('-'*40)
for i, c in enumerate(jogadores):
    print(f'{i:<3} {f'{jogadores[i]["Nome"]}':<10} {f'{jogadores[i]["Gol
s"]}':<20} {f'{jogadores[i]["Total de Gols"]}':<10}')
print('-'*40)
while True:
    esc = int(input('Mostrar dados de qual jogador ? (999 para parar ): '))
    if esc == 999:
        break
    while esc > len(jogadores)-1:
        esc = int(input(f'Jogador nº {esc} não existe! tente novamente ( 99
9 para parar ): '))
    print(f'  -- Levantamento do Jogador {jogadores[esc]["Nome"]}')
    for i, c in enumerate(jogadores[esc]['Gols']):
        print(f'No jogo {i+1} fez {c} gols')
print('Programa Finalizado')
```

Guanabara)
Bora entender o que está acontecendo neste código:

- declarado uma lista time onde vai receber todas as outras listas e dicionários, ela vai ser o local macro
- jogador vai ser um dicionário com todas as informações de cada jogador
- Partidas vai ser os valores de gols marcados em cada partida

Com isso vamos começar o código com um loop infinito

- dicionário jogador é limpo de todas as informações para adicionarmos o próximo
- key nome de jogador recebe um input
- tot é o input de quantas partidas o jogador jogou
- lista partidas é limpa para não transportar ou acumular as informações de inputs passados
- for c in range tot (ou seja ele vai repetir a quantidade de partidas que o jogador inseriu) recebe um input do valor dos gols de cada partida.
- key goals em jogador recebe a cópia dos dados da lista partidas
- key total recebe a soma da lista partidas
- time recebe o apend da cópia dos dados de jogador

Agora vamos para outro loop infinito para vermos se o usuário quer continuar ou não

- resp recebe um input se quer continuar ou não sendo levado para maiúscula e a primeira letra
- se resp tiver em 'SN' então ele faz um break para fora do loop senão ele dá um erro e volta
- se resp for 'n' então ele dá break e sai do loop principal

Vamos caminhando para a parte final do código

- print de ‘cod’ e end =’’ para colar na próxima linha
- varremos as chaves do dicionário mostrando ela num placeholder de 15 espaços com end = ’’ para ficarem na mesma linha
- damos um print vazio para dar espaço
- print de traços
- varremos a lista macro time com o enumerate com índice e valor, onde printamos o K sendo o indice num place holder de 4 espaços e end =’’
- verremos o v.values sendo o dicionário, usamos o for d in v.values e mostramos no print a string de d (pois temos alguns valores numéricos e outras listas) em 15 espaços e o end de ’’
- for desse for damos um print vazio para não juntar todas as linhas
- printamos linha para separar

Agora a última parte do código com o último while infinito

- busca recebe um int input perguntando qual jogador gostaria de saber o detalhamento
- se busca for 999 então ele para o código
- se busca for maior ou igual à len de time então ele dá erro e informa que não existe esse jogador
- senão ele fala o levantamento do time[indice sendo valor de busca][nome]
- vamos diretamente na lista que criamos em partidas que está dentro de jogador[’gols’], então fazemos o for i, g in enumerate time[busca][’gols’] e printamos um por um
- No final informamos que é o fim do programa

```
time = list()
jogador = dict()
partidas = list()

while True:
    jogador.clear()
    jogador['Nome'] = input('Nome do jogador: ')
    tot = int(input(f'Quantas partidas {jogador["Nome"]} jogou?: '))
    partidas.clear()
    for c in range(0,tot):
        partidas.append(int(input(f'  Quantos gols na partida {c}: ')))
    jogador['gols'] = partidas[:]
    jogador['total'] = sum(partidas)
    time.append(jogador.copy())
    while True:
        resp = input('Quer continuar [S/N]?: ').upper()[0]
        if resp in 'SN':
            break
        print('Erro! Responda apenas S ou N')
    if resp == 'N':
```

```
        break
print('cod ',end='')
for i in jogador.keys():
    print(f'{i:<15}',end='')
print()
print('-'*40)
for k, v in enumerate(time):
    print(f'{k:<4}',end='')
    for d in v.values():
        print(f'{str(d):<15}',end='')
    print()
print('-'*40)
while True:
    busca = int(input('Mostrar dados de qual jogador? (999 para parar): '))

    if busca == 999:
        break
    if busca >= len(time):
        print(f'ERRO! Não existe jogador com código {busca}')
    else:
        print(f'  ---- LEVANTAMENTO DO JOGADOR {time[busca]['Nome']}')
        for i, g in enumerate(time[busca]['gols']):
            print(f'    No jogo {i+1} fez {g} gols')
        print('-'*40)
print('Fim de programa')
```