Aula de Python – Semana 14/03/2022

- Correção dos Exercícios
- Novas formatações
- Funções de manipulação de strings

```python
## CORREÇÃO DOS EXERCÍCIOS DE ENTRADA, SAÍDA E PROCESSAMENTO DE DADOS

# ------------------------------------------------------------------------
# Exercício 1. Dado um numero pelo usuário, calcular o dobro
# Entrada: 6    Saída: 12
# - PASSO 1: Ler um número
num = input("Digite um numero: ")
# - PASSO 2: Calcular o dobro
num = float(num)
dobro = num * 2
# - PASSO 3: Exibir o Dobro
print(f"O dobro de {num} é {dobro}")

# ------------------------------------------------------------------------
# Exercício 2. Dado um numero pelo usuário, calcular o quadrado
# Entrada: 3    Saída: 9
# - PASSO 1: Ler um número
num = input("Digite um numero: ")
# - PASSO 2: Calcular o quadrado
num = float(num)
quadrado = num ** 2
# - PASSO 3: Exibir o Quadrado
print(f"O Quadrado de {num} é {quadrado:.1f}")

# ------------------------------------------------------------------------
# Exercício 3. Dado um numero pelo usuário, exibir o anterior e posterior
# Entrada: 23    Saída: 22 24
# - PASSO 1: Ler um número
num = input("Digite um numero: ")
num = int(num)
# - PASSO 2: Calcular o próximo
ant = num - 1
# - PASSO 3: Calcular o anterior
```

```
prox = num + 1
# - PASSO 4: Exibir o proxímo e o anterior
print(f"O número {num} tem como próximo o {prox} e como anterior o {ant}")

# ------------------------------------------------------------------------
# Exercício 4. Dados dois numeros pelo usuário, calcular a potencia
# Entrada: 2 4    Saída: 16
# - PASSO 1: Ler a Base
base = input("Digite a Base: ")
base = int(base)
# - PASSO 2: Ler o expoente
exp = input("Digite a Base: ")
exp = int(exp)
# - PASSO 3: Calcular a potencia
potencia = base ** exp
# - PASSO 4: Exibir a potencia
print(f"{base} elevado a {exp} = {potencia}")

# ------------------------------------------------------------------------
# Exercício 5. Dado um numero pelo usuário, exibir o proximo multiplo de 5
# Entrada: 8    Saída: 10
# Entrada: 15    Saída: 20
# Entrada: 1    Saída: 5
# Entrada: 89   Saída: 90
# ######### Observação: Resolver este exercício utilizando somente os comandos de
entrada, saída e processamento de dados
# - PASSO 1: Ler um número
num = input("Digite um numero: ")
# - PASSO 2: Calcular o próximo multiplo de 5
num = int(num)
prox_mult_5 = num // 5 * 5 + 5
# - PASSO 3: Exibir o Dobro
print(f"O próximo multiplo de 5 de {num} é {prox_mult_5}")
```

Conceitos novos

Comentando diversas linhas

Sabemos que o # efetua o comentário de uma linha.

Muitas vezes precisamos comentar diversas linhas e colocar o # em cada linha pode ser trabalhoso.

Para diminuir este trabalho, basta selecionar as linhas que deseja comentar e digitar simultaneamente:

<ctrl> + /

Novas formatações

Formatando valores com modificadores

:s – Texto (Strings)
:d – Inteiro (int)
:f – Números de Ponto Flutuante
:. – Casas decimais
:(Caractere) > Esquerda, < Direita ou ^ Centro

Utilizando o 'd' na formatação da variável inteira:

```
num = 23
print(f"Número {num:5d}")
print(f"Número {num:05d}")
print(f"Número {num:>05d}")
print(f"Número {num:<05d}")
print(f"Número {num:^05d}")
```

A saída:

```
Número    23
Número 00023
Número 00023
Número 23000
Número 02300
```

Utilizando o 's' na formatação da variável inteira:

```
nome = "Edson"
print(f"Nome = {nome:>10s}")
print(f"Nome = {nome:>010s}")
print(f"Nome = {nome:<10s}")
print(f"Nome = {nome:<010s}")
print(f"Nome = {nome:^10s}")
print(f"Nome = {nome:^010s}")
```

A saída:

```
Nome =      Edson
Nome = 00000Edson
Nome = Edson
Nome = Edson00000
Nome =   Edson
Nome = 00Edson000
```

Percebam a sintaxe da formatação:

`{<valor>:<caractere><operador><bytes>}`

Exemplos:

```
num = 23
print(f"Numero: {num:0>10}")
print(f"Numero: {num:0<10}")
print(f"Numero: {num:0^10}")
```

A saída destes exemplos:

```
Numero: 0000000023
Numero: 2300000000
Numero: 0000230000
```

Vejam o comportamento da formatação abaixo.

Nestes exemplos, as formatações estão ocorrendo diretamente nas variáveis, não dentro de um print.

```
nome = "Edson de Oliveira"
nome_formatado = '{:@>40}'.format(nome)
print(nome_formatado)
nome_formatado = '{:#<40}'.format(nome)
print(nome_formatado)
nome_formatado = '{:$^40}'.format(nome)
print(nome_formatado)
```

A saída destes exemplos:

```
@@@@@@@@@@@@@@@@@@@@@@@@Edson de Oliveira
Edson de Oliveira########################
$$$$$$$$$$Edson de Oliveira$$$$$$$$$$
```

Conceito de Função

Função é um tipo de instrução facilitadora que tem como objetivo atingir um resultado específico.

Neste momento do curso vamos nos ater somente às funções contidas no framework do Python. Em breve aprenderemos a construir as nossas funções.

Sem empregarmos o conceito acima acabamos utilizando a função format() que tem por objetivo formatar a saída dos dados.

Agora aprenderemos algumas funções que tratam strings”

Nos exemplos abaixo, atribuímos em outras variáveis o retorno das funções e depois usamos estas variáveis nos prints, veja:

```
nome = "Edson de Oliveira"
tamanho = len(nome) # Conta a qtd de caracteres da string
maiusculas = nome.upper() # Converte tudo em maiúsculo
minusculas = nome.lower() # Converte tudo em minúsculo
iniciais = nome.title() # Somente as iniciais em maiúsculo
print(f"{nome} tem {tamanho} caracteres")
print(f"Em maiúsculo: {maiusculas}")
print(f"Em minúsculo: {minusculas}")
print(f"Iniciais: {iniciais}")
```

A saída destes exemplos:

```
Edson de Oliveira tem 17 caracteres
Em maiúsculo: EDSON DE OLIVEIRA
Em minúsculo: edson de oliveira
Iniciais: Edson De Oliveira
```

Nos exemplos abaixo obtivemos os mesmos resultados (saídas) acima, mas não utilizamos variáveis auxiliares, ou seja, executamos as funções diretamente nos prints. Vejam:

```
nome = "Edson de Oliveira"
print(f"{nome} tem {len(nome)} caracteres")
print(f"Em maiúsculo: {nome.upper()}")
print(f"Em minúsculo: {nome.lower()}")
print(f"Iniciais: {nome.title()}")
```

Qual a diferença entre estas duas formas?

Na visão do usuário, não há diferença porque a saída é a mesma, mas para o programador sim.

Quando atribuímos em uma variável, podemos recuperar esta informação posteriormente. Agora, se usamos diretamente num print esta informação se perde e não pode ser utilizada no decorrer do programa.