

ALGORITMIA E ESTRUTURAS DE DADOS

MÓDULO V TKINTER — PARTE II

TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB

A BIBLIOTECA TKINTER

- ☐ Menu *drop-down / pull-down*
- ☐ Message
- ☐ MessageBox
- ☐ TreeView

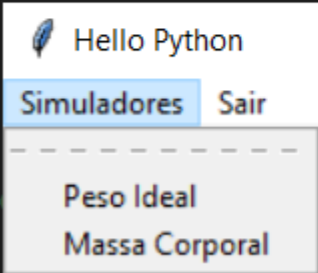
❖ Biblioteca Tkinter

❑ **Menu** – widget que permite implementar menus drop-down / pull-down

```
4
5  window=Tk()    # invoca classe tk , cria a "main window"
6  window.geometry("500x300")
7  window.title('Hello Python')
8
9  # Implementar menu
10 barra_Menu = Menu(window)
11
12 # Constroi menu Simuladores, com 2 opções drop-down
13 simuladores_Menu = Menu(barra_Menu) # objeto associado à barra de menu
14 simuladores_Menu.add_command(label = "Peso Ideal", command= "noaction")
15 simuladores_Menu.add_command(label = "Massa Corporal", command= "noaction")
16 barra_Menu.add_cascade(label = "Simuladores", menu = simuladores_Menu)
17
18 # Constroi menu Sair, com comando quit
19 barra_Menu.add_command(label = "Sair", command = window.quit)
20
21 window.configure(menu=barra_Menu)
22
23
24 window.mainloop() # event listening 1
```

Objeto que irá conter todo o menu

Menu Simuladores



❖ Treeview

- ❑ Componente que permite apresentar dados em formato agregado, de tabela
- ❑ Pode-se considerar que é uma forma visual de implementar listas bidimensionais (linhas e colunas)
- ❑ Componente muito usado em aplicações, nomeadamente em consultas de dados

The screenshot shows a software application window titled "Consultas". On the left, there are two filter sections. The first, "Tipo de Movimento", has two checked checkboxes: "Entrada" and "Saída". The second, "Por Utilizador", has a text input field labeled "Número:" and a "Consultar" button below it. At the bottom left is a button labeled "Obter Dados". On the right, a table displays the results of the query. The table has four columns: "Número", "Data", "Hora", and "Movimento". It contains 12 rows of data. At the bottom right, there are four more input fields labeled "Número:", "Data:", "Hora:", and "Movimento:".

| Número | Data | Hora | Movimento |
|--------|------------|-----------------|-----------|
| 220 | 2021-12-29 | 16:00:38.329945 | Entrada |
| 220 | 2021-12-29 | 16:00:39.838099 | Saída |
| 120 | 2021-12-29 | 16:00:43.884554 | Entrada |
| 120 | 2021-12-29 | 16:00:45.185505 | Saída |
| 178 | 2022-01-05 | 14:53:39.240774 | Entrada |
| 220 | 2021-12-30 | 16:00:38.329945 | Entrada |
| 220 | 2021-12-30 | 16:00:39.838099 | Saída |
| 120 | 2021-12-30 | 16:00:43.884554 | Entrada |
| 120 | 2021-12-30 | 16:00:45.185505 | Saída |
| 178 | 2022-01-07 | 14:53:39.240774 | Entrada |

❖ Treeview

- ❑ Componente do módulo ttk (que deve ser importado)

```
Ex01.py > ...  
1  # Biblioteca Tkinter: UI  
2  from tkinter import *  
3  from tkinter import ttk # treeview  
4  from tkinter import messagebox  
5  from PIL import ImageTk, Image  
6
```

1. Definição da colunas do componente Treeview:

Mostrar cabeçalhos:

- headings
- tree, tree headings
- "

Nomes das colunas (nome interno, no código)

```
# Paine1 2  
panel2 = PanedWindow(conWindow, width = 450, height = 270, bd = "3", relief = "sunken")  
panel2.place(x=220, y=20)  
# TreeView para consulta de movimentos  
global tree  
tree = ttk.Treeview(panel2, selectmode = "browse", columns = ("Número", "Data", "Hora", "Movimento"), show = "headings")  
  
tree.column("Número", width = 100, anchor="c")  
tree.column("Data", width = 100, anchor="c")  
tree.column("Hora", width = 100, anchor="c")  
tree.column("Movimento", width = 140, anchor="c")  
tree.heading("Número", text = "Número")  
tree.heading("Data", text = "Data")  
tree.heading("Hora", text = "Hora")  
tree.heading("Movimento", text = "Movimento")  
tree.place(x=5, y=5)
```

c- center, e - direita, w- esquerda

❖ Treeview

1. Definição da colunas do componente Treeview:

Propriedade **column**: para personalizar cada coluna da Treeview

```
# Paine1 2
panel2 = PanedWindow(conWindow, width = 450, height = 270, bd = "3", relief = "sunken")
panel2.place(x=220, y=20)
# TreeView para consulta de movimentos
global tree
tree = ttk.Treeview(panel2, selectmode = "browse", columns = ("Número", "Data", "Hora", "Movimento"), show = "headings")

tree.column("Número", width = 100, anchor="c")
tree.column("Data", width = 100, anchor="c")           # c- center, e - direita, w- esquerda
tree.column("Hora", width = 100, anchor="c")
tree.column("Movimento", width = 140, anchor="c")
tree.heading("Número", text = "Número")
tree.heading("Data", text = "Data")
tree.heading("Hora", text = "Hora")
tree.heading("Movimento", text = "Movimento")
tree.place(x=5, y=5)
```

Propriedade **heading**: para personalizar cada cabeçalho da Treeview

❖ Treeview

2. Adicionar dados ao componente Treeview: **insert**

```
tree.insert("", "end", values = (campos[0], campos[1], campos[2], campos[3]))
```

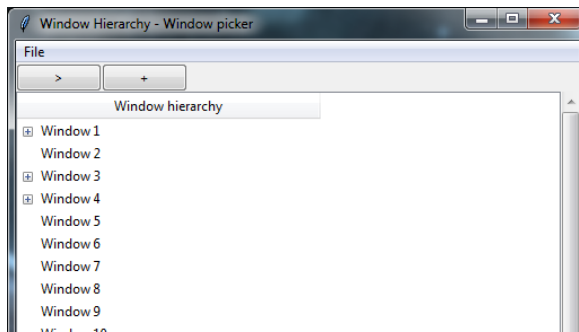
Por omissão, vazio.
Uma **Treeview** permite
inserir linhas com diferentes
níveis de indentação
(hierárquicos).

Nesse caso, este parâmetro
indica qual o nível
hierárquico a inserir

Dados inserir na Treeview, pela ordem
de definição das colunas

Índice:

- 0 (primeira linha)
- número inteiro, da linha a inserir na Treeview
- end (no final da Treeview)



❖ Treeview

2. Adicionar dados ao componente Treeview: **insert**

```
def dados_treeview():  
    tree.delete(*tree.get_children()) # Remove todas as linhas da Treeview  
    mov = ""  
    if cb1.get() == True and cb2.get() == True:  
        mov = "T" # Filtrar por TODOS os movimentos  
    else:  
        if cb1.get() == True:  
            mov = "Entrada\n" # Filtrar por Entradas  
        if cb2.get() == True:  
            mov = "Saída\n" # Filtrar por saídas  
    lista = ler_ficheiro() # ler Conteúdo do ficheiro de presencas.txt  
    for linha in lista:  
        campos = linha.split(";")  
        if mov == "T" or campos[3] == mov:  
            if utilizador.get() == "" or utilizador.get() == campos[0]:  
                tree.insert("", "end", values = (campos[0], campos[1], campos[2], campos[3]))
```


❖ Treeview

2. Obter dados da linha ativa/selecionada, da Treeview

```
def dados_linha_tree():  
    """  
    Renderiza nas Entry os dados da linha selecionada  
    """  
    row_id = tree.focus() # obter o id da linha ativa / selecionada  
    linha = tree.item(row_id)  
    value_num.set(linha["values"][0])  
    value_data.set(linha["values"][1])  
    value_hora.set(linha["values"][2])  
    value_mov.set(linha["values"][3])
```

Consultas

Tipo de Movimento

☒ Entrada

☒ Saída

Por Utilizador

Número:

Consultar

| Número | Data | Hora | Movimento |
|--------|------------|-----------------|-----------|
| 220 | 2021-12-29 | 16:00:38.329945 | Entrada |
| 220 | 2021-12-29 | 16:00:39.838099 | Saída |
| 120 | 2021-12-29 | 16:00:43.884554 | Entrada |
| 120 | 2021-12-29 | 16:00:45.185505 | Saída |
| 178 | 2022-01-05 | 14:53:39.240774 | Entrada |
| 220 | 2021-12-30 | 16:00:38.329945 | Entrada |
| 220 | 2021-12-30 | 16:00:39.838099 | Saída |
| 120 | 2021-12-30 | 16:00:43.884554 | Entrada |
| 120 | 2021-12-30 | 16:00:45.185505 | Saída |
| 178 | 2022-01-07 | 14:53:39.240774 | Entrada |

Obter Dados

Número: 178

Data: 2022-01-05

Hora: 14:53:39.240

Movimento: Entrada

❖ Treeview

3. Remover dados ao componente Treeview: **delete**

```
def remove_tree():  
    tree.delete(*tree.get_children()) # REMOVE todas as linhas da tree  
    # OU:  
    linhas = tree.get_children() # Obter todas as linhas  
    for linha in linhas: # remove uma a uma  
        tree.delete(linha)
```

Consultas

Tipo de Movimento

☒ Entrada
☐ Saída

Por Utilizador

Número:

Consultar

| Número | Data | Hora | Movimento |
|--------|------------|-----------------|-----------|
| 220 | 2021-12-29 | 16:00:38.329945 | Entrada |
| 120 | 2021-12-29 | 16:00:43.884554 | Entrada |
| 178 | 2022-01-05 | 14:53:39.240774 | Entrada |
| 220 | 2021-12-30 | 16:00:38.329945 | Entrada |
| 120 | 2021-12-30 | 16:00:43.884554 | Entrada |

Obter Dados

Remove Tudo

Número:
Data:
Hora:
Movimento:

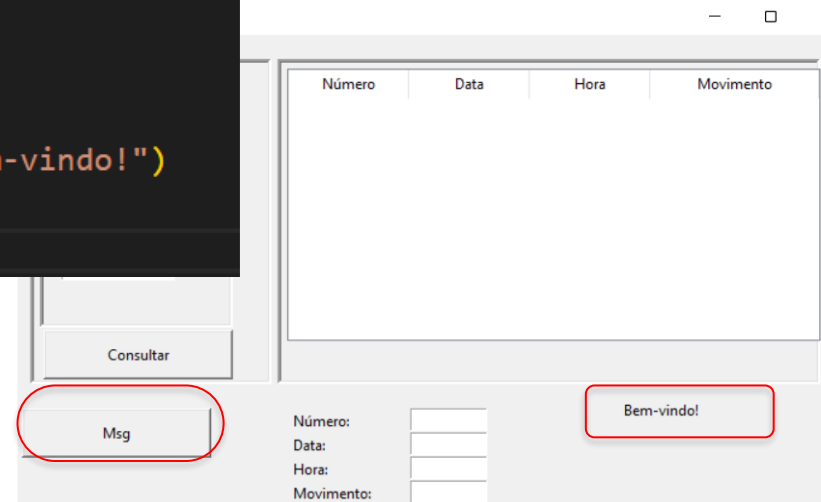
❖ Biblioteca Tkinter

❑ **Message** – componente para imprimir mensagem

- ❑ bg, fg, bd, text, font, width, height
- ❑ wraplength (nº de caracteres a imprimir por linha; default 0, muda de linha de acordo com \n)
- ❑ anchor (default anchor = center)

```
btn_mensagem = Button(ConWindow, width = 21, height= 2, text = "Msg", relief = "raised", command = mensagem)  
btn_mensagem.place(x=8, y=310)
```

```
def mensagem():  
    """  
    renderiza mensagem ao utilizador  
    """  
    msg = Message(ConWindow, width = 100, text = "Bem-vindo!")  
    msg.place(x=500, y= 300)
```



❖ Biblioteca Tkinter

- ❑ **Message** – componente para imprimir mensagem
 - ❑ bg, fg, bd, text, font, width, height
 - ❑ wraplength (nº de caracteres a imprimir por linha; default 0, muda de linha de acordo com \n)
 - ❑ anchor (default anchor = center)

```
texto= "Bem-Vindo!"  
btn_mensagem = Button(ConWindow, width = 21, height= 2, text = "Msg", relief = "raised",  
                        command = lambda: mensagem(texto))  
btn_mensagem.place(x=8, y=310)
```

```
def mensagem(texto):  
    """  
    renderiza mensagem ao utilizador  
    """  
    msg = Message(ConWindow, width = 100, text = texto)  
    msg.place(x=500, y= 300)
```

lambda:

Funções anónimas cujo principal objetivo é o de criar um argumento para outra função o receber

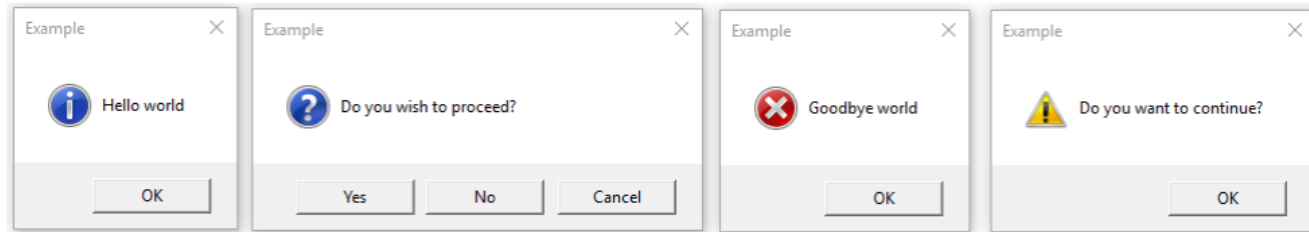
O conteúdo do argumento fica Temporariamente em memória

❖ Biblioteca Tkinter

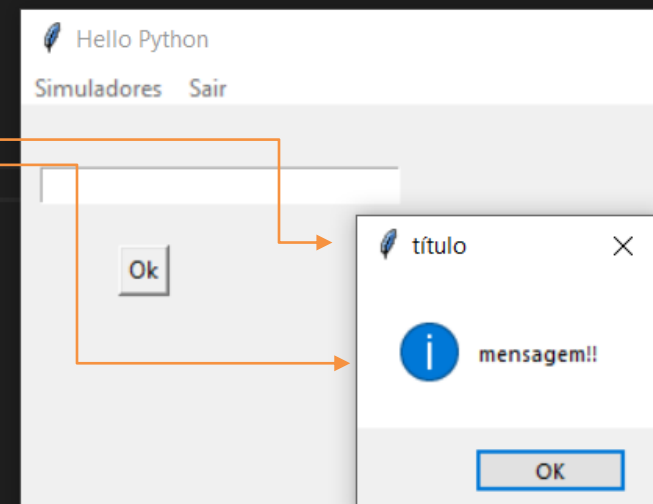
```
exemplo3.py > ...  
1  # Biblioteca Tkinter: UI  
2  from tkinter import *  
3  from tkinter import messagebox  
4
```

❑ **messagebox** – o módulo *messagebox* permite a implementação de mensagens em caixas de diálogo. Este módulo providencia diversas funções para personalizar as mensagens, tais como:

❑ `showinfo`, `showwarning`, `showerror`, `askquestion`, `askokcancel`, `askyesno`, `askretryignore`

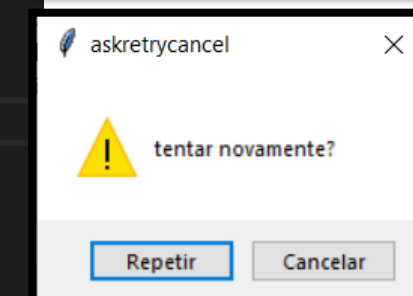
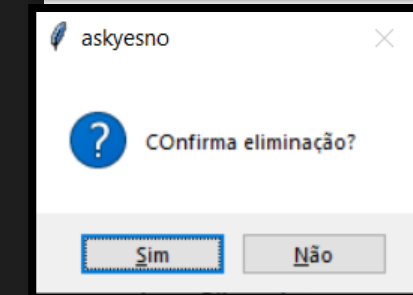
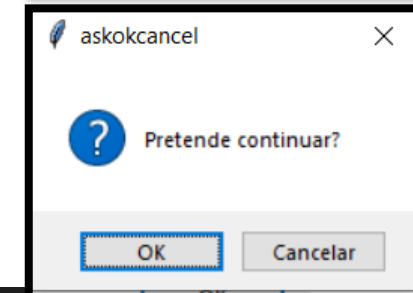
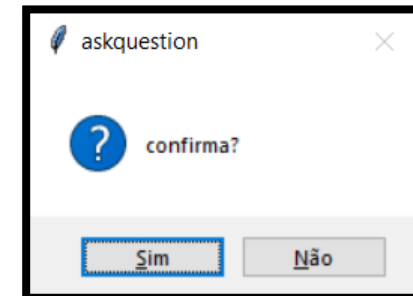
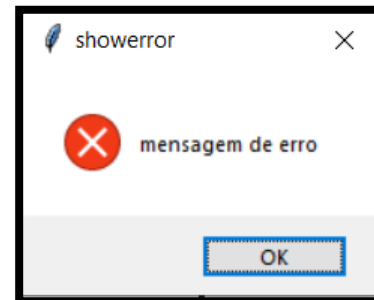
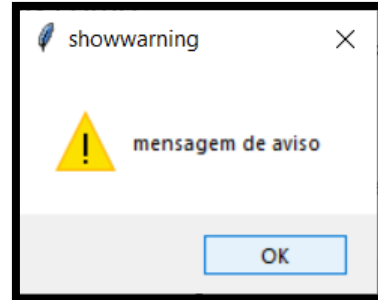
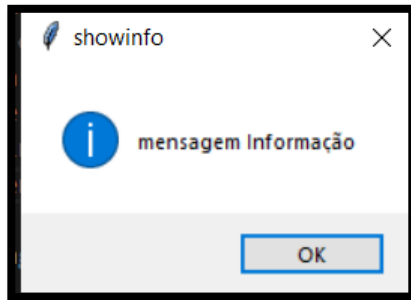


```
25  
26 txt_texto = Entry(window, width = 30, relief = "sunken")  
27 txt_texto.place(x=10, y=30)  
28  
29 def mensagem():  
30     #MessageBox  
31     messagebox.showinfo(title = "título", message = "mensagem!!")  
32  
33  
34 btn = Button(window, text = "Ok" , command = mensagem)  
35 btn.place(x=50, y=70)  
36  
37  
38  
39  
40  
41
```



❖ Biblioteca Tkinter

❑ messagebox



```
def mensagem():  
    #MessageBox  
    messagebox.showinfo("showinfo", "mensagem Informação")  
    messagebox.showwarning("showwarning", "mensagem de aviso")  
    messagebox.showerror("showerror", "mensagem de erro")  
    messagebox.askquestion("askquestion", "confirma?")  
    messagebox.askokcancel("askokcancel", "Pretende continuar?")  
    messagebox.askyesno("askyesno", "Confirma eliminação?")  
    messagebox.askretrycancel("askretrycancel", "tentar novamente?")
```

Título caixa

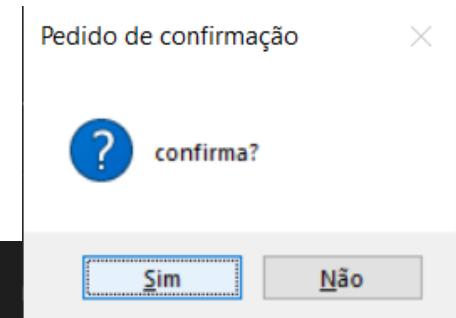
texto da mensagem

```
btn = Button(window, text = "Ok" , command = mensagem)  
btn.place(x=50, y=70)
```

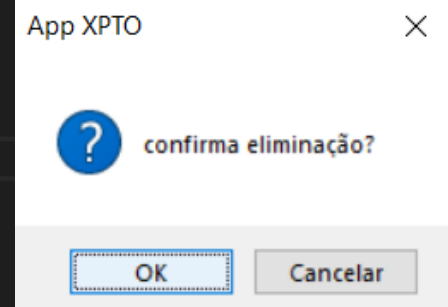
❖ Biblioteca Tkinter

❑ messagebox

```
p1.py > ...  
1  from tkinter import messagebox  
2  
3  res = messagebox.askquestion("Pedido de confirmação", "confirma?")  
4  if res == "yes":  
5      messagebox.showinfo("info", "a resposta foi Sim")  
6  elif res == "no":  
7      messagebox.showinfo("info", "a resposta foi Não")  
8  
9
```



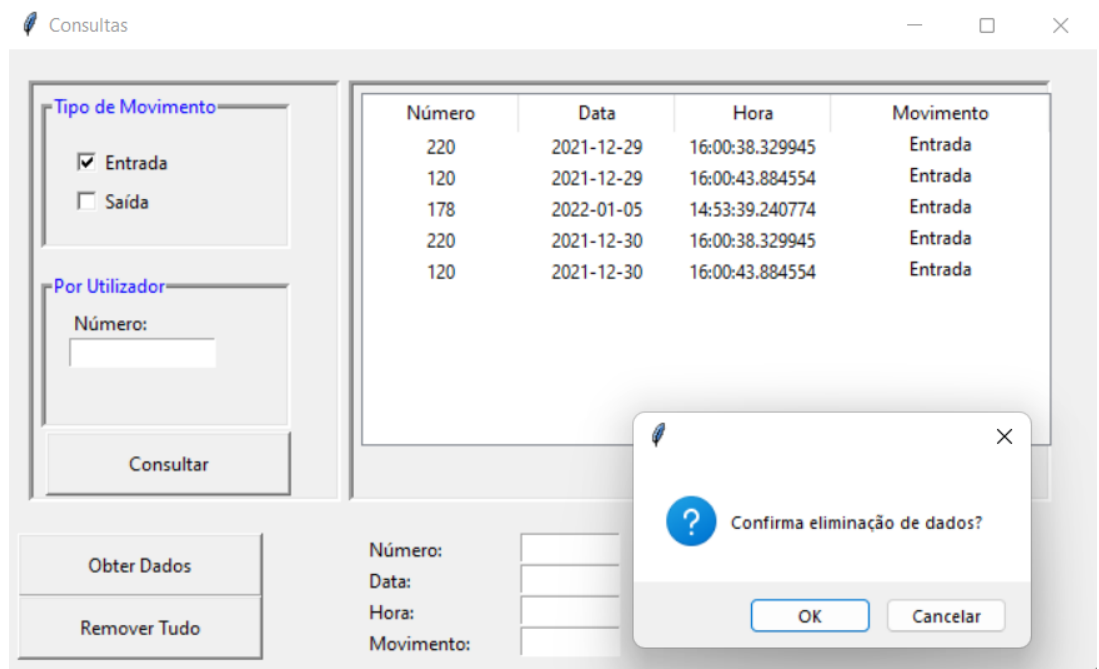
```
p1.py > ...  
1  from tkinter import messagebox  
2  
3  
4  res = messagebox.askokcancel("App XPTO", "confirma eliminação?")  
5  if res == True:  
6      messagebox.showinfo("info", "a resposta foi Ok")  
7  elif res == False:  
8      messagebox.showinfo("info", "a resposta foi Cancel")
```



❖ Biblioteca Tkinter

❑ messagebox

```
6
7 def remove_tree():
8     res= messagebox.askokcancel(title= "", message="Confirma eliminação de dados?")
9     if res == True:
10         tree.delete(*tree.get_children()) # REMOVE todas as linhas da tree
11
12
```



❖ Biblioteca Tkinter

❑ messagebox

```
def remove_tree():  
    res= messagebox.askquestion(title= "", message="Confirma eliminação de dados?")  
    if res == "yes":  
        tree.delete(*tree.get_children()) # REMOVE todas as linhas da tree
```



Consultas

Tipo de Movimento

☒ Entrada
☐ Saída

Por Utilizador

Número:

Consultar

| Número | Data | Hora | Movimento |
|--------|------------|-----------------|-----------|
| 220 | 2021-12-29 | 16:00:38.329945 | Entrada |
| 120 | 2021-12-29 | 16:00:43.884554 | Entrada |
| 178 | 2022-01-05 | 14:53:39.240774 | Entrada |
| 220 | 2021-12-30 | 16:00:38.329945 | Entrada |
| 120 | 2021-12-30 | 16:00:43.884554 | Entrada |

Obter Dados

Remover Tudo

Número:
Data:
Hora:
Movimento:

Confirma eliminação de dados?

Sim Não