

CC8210 – NCA210

Programação Avançada I

Prof. Reinaldo A. C. Bianchi
Prof. Isaac Jesus da Silva
Prof. Danilo H. Perico

Aula de Hoje

- Introdução à Interfaces Gráficas – GUI.
- Referências:
 - <https://docs.python.org/3/library/tk.html>

Python

GUI - ***G***raphical ***U***ser ***I***nterface

GUI - *Graphical User Interface*

- Interface Gráfica do Usuário
- Por que utilizar *Interface Gráfica do Usuário*?
- Foco no usuário final:
 - Mais amigável
 - Interação mais rápida
 - Mais produtiva

GUI - Frameworks para Python

- Tkinter (Tk interface)
- PyQt
- wxPython
- etc

GUI - Frameworks para Python

- **Tkinter (Tk interface):**
 - Toolkit padrão do Python para desenvolvimento de GUI
- PyQt
- wxPython
- etc

GUI - Frameworks para Python

- Vamos usar o **Tkinter**:
 - O **Tkinter** é um pacote interessante que já vem por padrão com o Python.
 - Para usar o Tkinter, só precisamos importá-lo:

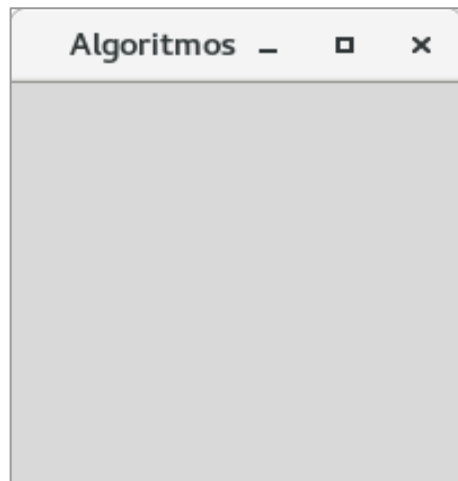
```
from tkinter import *
```

- O Tkinter permite a criação de *janelas*, *rótulos*, *botões*, *caixas de texto*, *caixas de mensagem* etc.

GUI - Janela

Cria a janela gráfica

```
from tkinter import *  
  
# cria a janela  
janela = Tk()  
  
# titulo para a janela  
janela.title("Algoritmos")  
  
# configura o tamanho da janela  
janela.geometry('400x400')  
  
# chama a função mainloop:  
# loop infinito para manter a janela aberta  
janela.mainloop()
```



GUI - Janela

```
from tkinter import *
```

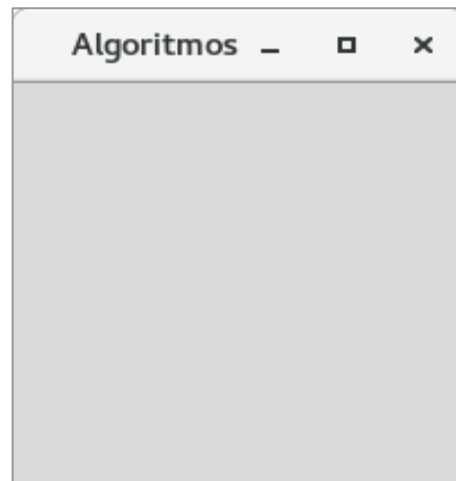
```
# cria a janela  
janela = Tk()
```

```
# titulo para a janela  
janela.title("Algoritmos")
```

```
# configura o tamanho da janela  
janela.geometry('400x400')
```

```
# chama a função mainloop:  
# loop infinito para manter a janela aberta  
janela.mainloop()
```

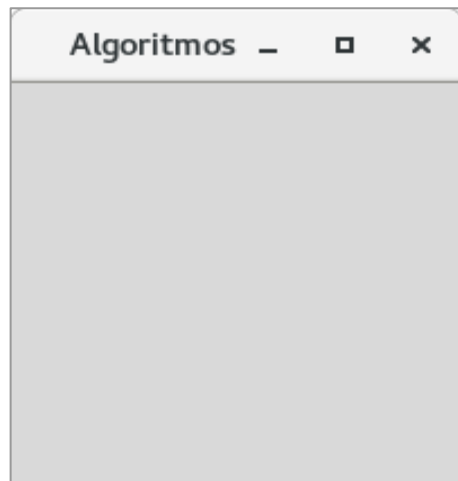
Define o título da
janela gráfica



GUI - Janela

```
from tkinter import *  
  
# cria a janela  
janela = Tk()  
  
# titulo para a janela  
janela.title("Algoritmos")  
  
# configura o tamanho da janela  
janela.geometry('400x400')  
  
# chama a função mainloop:  
# loop infinito para manter a janela aberta  
janela.mainloop()
```

Define o tamanho da
janela gráfica



GUI - Janela

```
from tkinter import *
```

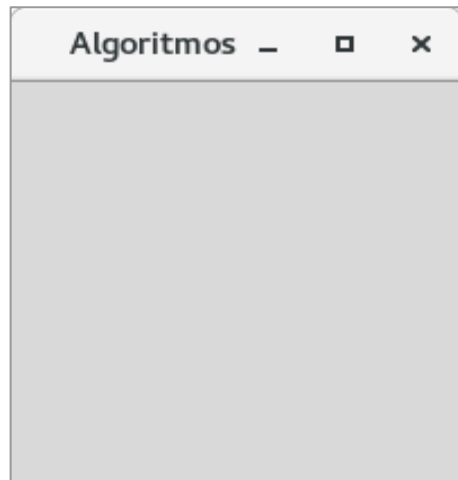
```
# cria a janela  
janela = Tk()
```

```
# titulo para a janela  
janela.title("Algoritmos")
```

```
# configura o tamanho da janela  
janela.geometry('400x400')
```

```
# chama a função mainloop:  
# loop infinito para manter a janela aberta  
janela.mainloop()
```

IMPORTANTE:
Esta função que
inicia e mantém a
janela aberta!!



GUI - Rótulo (*Label*)

```
from tkinter import *

# cria a janela
janela = Tk()

# titulo para a janela
janela.title("Algoritmos")

# configura o tamanho da janela
janela.geometry('400x400')

# cria o rótulo na janela desejada, com o texto desejado e configura a fonte
rotulo = Label(janela, text="Primeira aplicação gráfica no Python!", font=("Arial Bold", 14))

# configura onde o texto vai aparecer na janela:
# x = 200 e y = 100
# a referência é o centro (CENTER) do rótulo
rotulo.place(x=200, y=100, anchor=CENTER)

# chama a função mainloop:
# loop infinito para manter a janela aberta
janela.mainloop()
```



GUI - Posicionamento dos Elementos (*place*)

- O *place* permite que os elementos sejam explicitamente posicionados de forma absoluta ou relativa
- Sintaxe:

- Posicionando o elemento w, de forma relativa (centralizando):

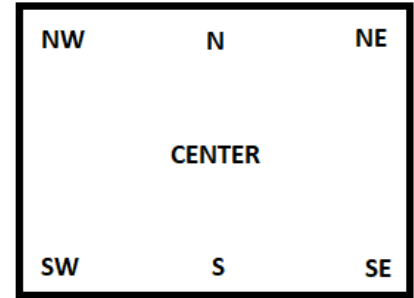
```
w.place(relx=0.5, rely=0.5, anchor=CENTER)
```

- Posicionando o elemento w de forma absoluta:

```
w.place(x = 50, y = 100, anchor=CENTER)
```

GUI - Tkinter *place*

- *anchor* refere-se ao elemento que está sendo posicionado:
 - Pode assumir os valores:
 - NW (default), N, NE, E, SE, S, SW, W, CENTER



```
from tkinter import *  
  
window = Tk()  
  
window.title("Algoritmos")  
  
window.geometry('350x200')  
  
btn = Button(window, text="Clique!")  
btn.place(relx = 0.5, rely = 0.5, anchor=CENTER)  
  
btn2 = Button(window, text="Clique2!")  
btn2.place(x = 100, y = 50, anchor=CENTER)  
  
window.mainloop()
```



GUI - Botão (*Button*)

```
from tkinter import *

# cria a janela
janela = Tk()

# titulo para a janela
janela.title("Algoritmos")

# configura o tamanho da janela
janela.geometry('400x400')

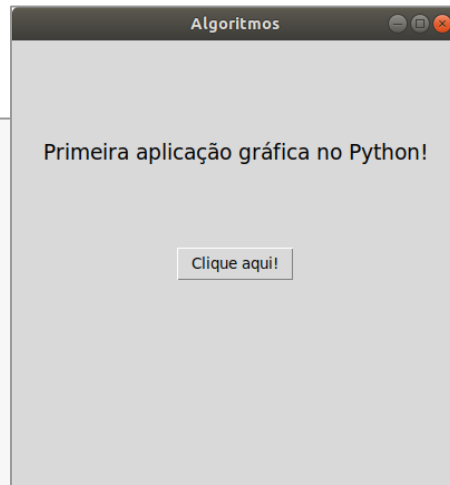
# cria o rótulo na janela desejada, com o texto desejado e configura a fonte
rotulo = Label(janela, text="Primeira aplicação gráfica no Python!", font=("Arial Bold", 14))

# configura onde o texto vai aparecer na janela:
# x = 200 e y = 100
# a referência é o centro (CENTER) do rótulo
rotulo.place(x=200, y=100, anchor=CENTER)

# cria o botão na janela desejada, com o texto desejado
botao = Button(janela, text="Clique aqui!")

# configura onde o botão vai aparecer na janela
botao.place(x=200, y=200, anchor=CENTER)

# chama a função mainloop:
# loop infinito para manter a janela aberta
janela.mainloop()
```



GUI - Botão (*Button*)

- O botão criado no slide anterior **não tem nenhuma utilidade!**
- Normalmente, o **clique de um botão é associado** a chamada de uma **função!**

GUI - Botão (*Button*)

```
# configura onde o texto vai aparecer na janela:
# x = 200 e y = 100
# a referência é o centro (CENTER) do rótulo
rotulo.place(x=200, y=100, anchor=CENTER)

# definição da função clique()
def clique():
    rotulo['text'] = "Novo texto!"

# cria o botão na janela desejada, com o texto desejado e estabelece
# qual a função que será chamada no clique
botao = Button(janela, text="Clique aqui!", command=clique)

# configura onde o botão vai aparecer na janela
botao.place(x=200, y=200, anchor=CENTER)

# chama a função mainloop:
# loop infinito para manter a janela aberta
janela.mainloop()
```

O texto foi alterado depois do clique!



GUI - Caixas de Texto (*Entry*)

```
# configura onde o texto vai aparecer na janela:
# x = 200 e y = 100
# a referência é o centro (CENTER) do rótulo
rotulo.place(x=200, y=100, anchor=CENTER)

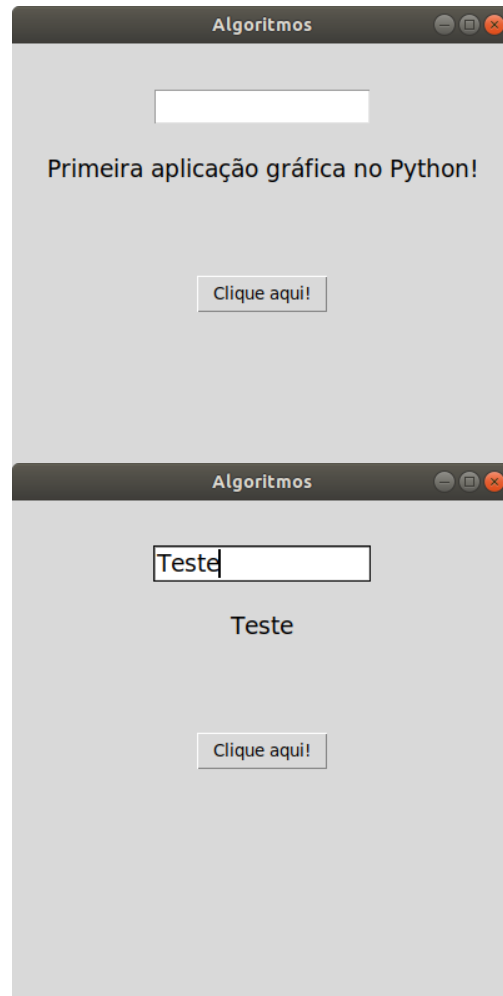
# cria o elemento de entrada de texto, configura o tamanho
entrada = Entry(janela, width=14, font=("Arial Bold", 14))
entrada.place(x=200, y=50, anchor=CENTER)

# definição da função clique()
def clique():
    resposta = entrada.get()
    rotulo['text'] = resposta

# cria o botão na janela desejada, com o texto desejado e estabelece
# qual a função que será chamada no clique
botao = Button(janela, text="Clique aqui!", command=clique)

# configura onde o botão vai aparecer na janela
botao.place(x=200, y=200, anchor=CENTER)

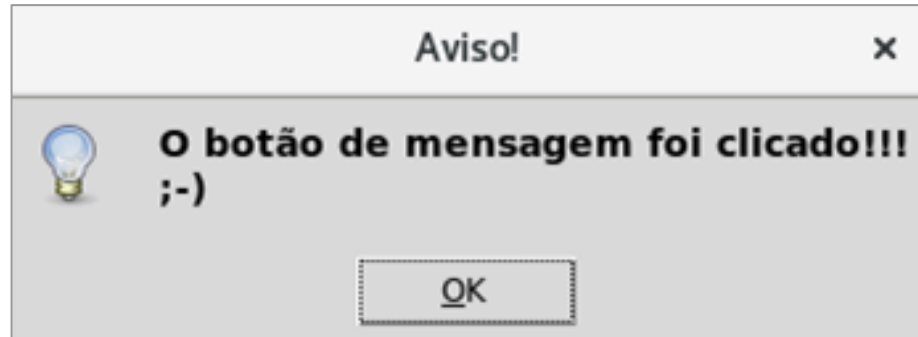
# chama a função mainloop:
# loop infinito para manter a janela aberta
janela.mainloop()
```



GUI - Caixas de Mensagem (*MessageBox*)

```
from tkinter import *  
from tkinter import messagebox
```

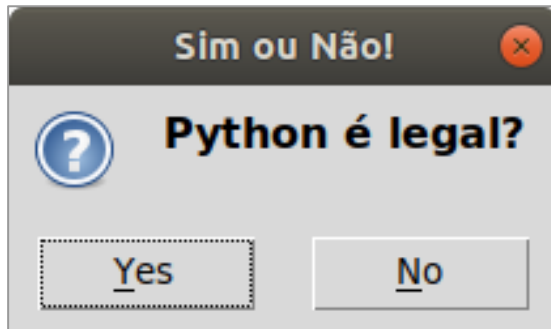
```
def show():  
    res = messagebox.showinfo('Aviso', 'O botão de mensagem foi clicado!!! ;~)')  
    print(res)  
  
botao2 = Button(janela, text='Mensagem', command=show)
```



GUI - Caixas de Mensagem (*MessageBox*)

- Caixas de diálogo com perguntas:

```
def show():  
    res = messagebox.askyesno('Sim ou Não!', 'Python é legal?')  
    print(res)
```



- **res** terá valor **True** ou **False**, dependendo do botão que for clicado.

GUI - Caixas de Mensagem (*MessageBox*)

- Caixas de diálogo com perguntas / respostas:

```
res = messagebox.askyesno('Aviso!', 'O botão de mensagem foi clicado!!! ;-)')
res = messagebox.askquestion('Aviso!', 'O botão de mensagem foi clicado!!! ;-)')
res = messagebox.askyesnocancel('Aviso!', 'O botão de mensagem foi clicado!!! ;-)')
res = messagebox.askokcancel('Aviso!', 'O botão de mensagem foi clicado!!! ;-)')
res = messagebox.askretrycancel('Aviso!', 'O botão de mensagem foi clicado!!! ;-)')
```

GUI - Caixas de Mensagem (*MessageBox*)

Information message box

```
tkinter.messagebox.showinfo(title=None, message=None, **options)
```

Warning message boxes

```
tkinter.messagebox.showwarning(title=None, message=None, **options)
```

```
tkinter.messagebox.showerror(title=None, message=None, **options)
```

Question message boxes

```
tkinter.messagebox.askquestion(title=None, message=None, **options)
```

```
tkinter.messagebox.askokcancel(title=None, message=None, **options)
```

```
tkinter.messagebox.askretrycancel(title=None, message=None, **options)
```

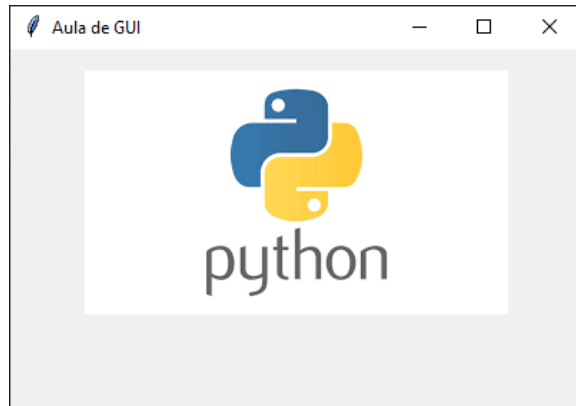
```
tkinter.messagebox.askyesno(title=None, message=None, **options)
```

```
tkinter.messagebox.askyesnocancel(title=None, message=None, **options)
```

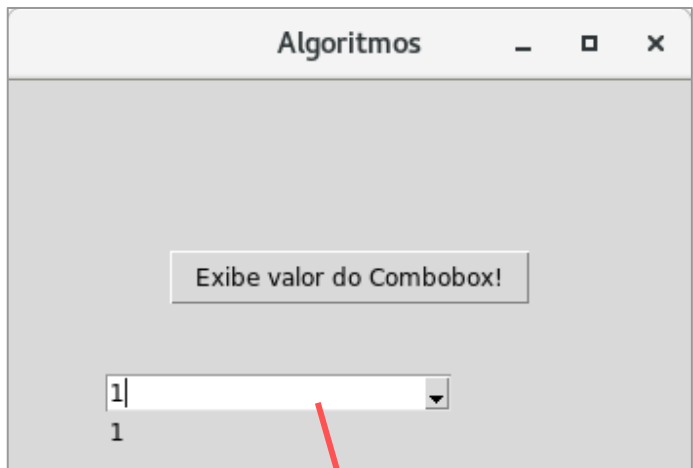
GUI – Incluindo imagens

- Podemos adicionar imagens usando Label:

```
imagem = PhotoImage(file="images.png")  
w = Label(janela, image=imagem)  
w.place(x=200, y=220, anchor=CENTER)
```

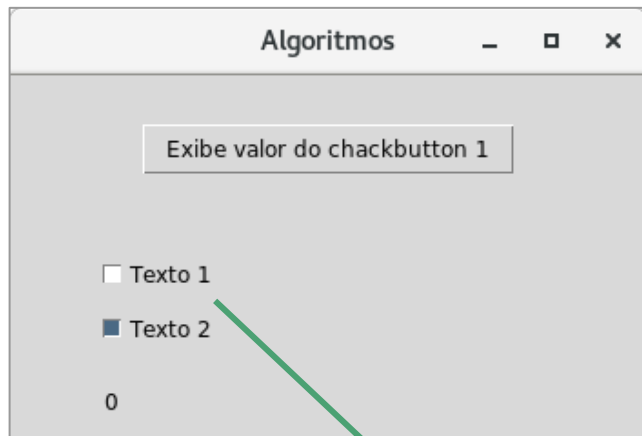


GUI - Caixas de Combinação (*Combobox*)



```
1 from tkinter import *
2 from tkinter.ttk import Combobox
3
4 window = Tk()
5 window.title("Algoritmos")
6 window.geometry('350x200')
7
8 # cria o combobox
9 combobox = Combobox(window)
10 # insere os possíveis valores
11 combobox['values'] = (1, 2, 3, 4, 5, "Text")
12 # configura o item selecionado
13 combobox.current(2)
14 # posiciona o combobox
15 combobox.place(x = 50, y = 150)
16
17 def show():
18     # para ler o valor selecionado
19     label['text'] = combobox.get()
20
21 btn = Button(window, text="Exibe valor do Combobox!", command=show)
22 btn.place(relx = 0.5, rely = 0.5, anchor=CENTER)
23
24 label = Label(window)
25 label.place(x=50, y=170)
26
27 window.mainloop()
```


GUI - Botão de Verificação (*Checkbutton*)

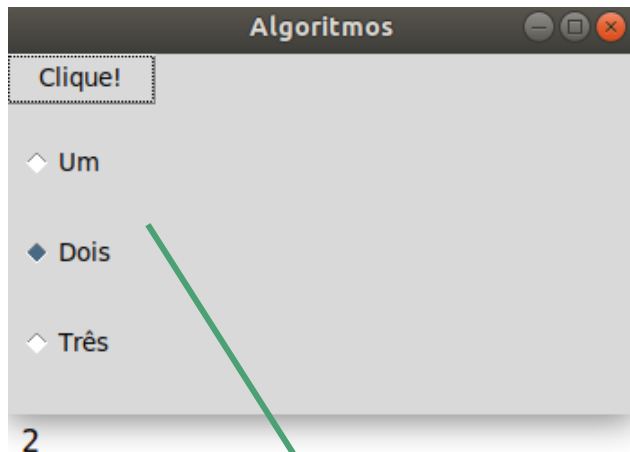


checkbutton

- Checkbutton selecionado:
 - `chk_state.get() == 1`
- Não selecionado:
 - `chk_state.get() == 0`

```
1 from tkinter import *
2 from tkinter.ttk import Checkbutton
3
4 window = Tk()
5 window.title("Algoritmos")
6 window.geometry('350x200')
7
8 #cria a variável de estado
9 chk_state = BooleanVar()
10 #configura o estado
11 chk_state.set(False)
12 #cria a variável de estado
13 chk_state2 = BooleanVar()
14 #configura o estado
15 chk_state2.set(True)
16 # cria o checkbutton
17 chk = Checkbutton(window, text='Texto 1', var=chk_state)
18 chk2 = Checkbutton(window, text='Texto 2', var=chk_state2)
19 chk.place(x = 50, y = 100)
20 chk2.place(x = 50, y = 130)
21
22 def show():
23     label['text'] = chk_state.get()
24
25 btn = Button(window, text="Exibe valor do chackbutton 1", command=show)
26 btn.place(relx = 0.5, rely = 0.2, anchor=CENTER)
27
28 label = Label(window)
29 label.place(x=50, y=170)
30
31 window.mainloop()
```

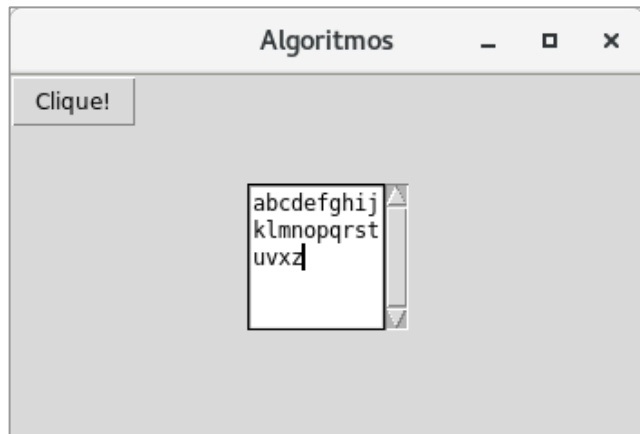
GUI - Botão de Opção (*Radiobutton*)



radiobutton

```
1 from tkinter import *
2 from tkinter.ttk import *
3
4 window = Tk()
5 window.title("Algoritmos")
6 window.geometry('350x200')
7
8 selected = IntVar()
9
10 rad1 = Radiobutton(window, text='Um', value=1, variable=selected)
11 rad2 = Radiobutton(window, text='Dois', value=2, variable=selected)
12 rad3 = Radiobutton(window, text='Três', value=3, variable=selected)
13
14 rad1.place(x=10, y=50)
15 rad2.place(x=10, y=100)
16 rad3.place(x=10, y=150)
17
18 def clicked():
19     print(selected.get())
20
21 btn = Button(window, text="Clique!", command=clicked)
22 btn.grid(column=0, row=3)
23
24 window.mainloop()
```

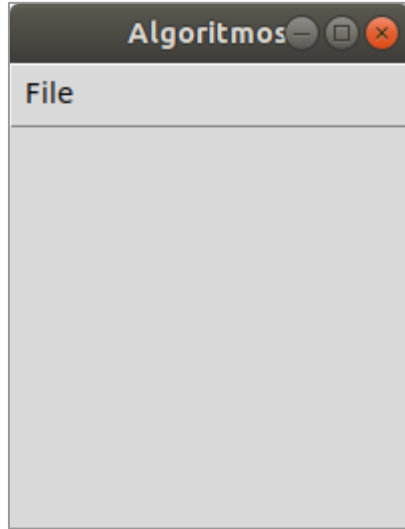
GUI - Área de texto com barra de rolagem (*ScrolledText*)



```
1 from tkinter import *
2 from tkinter import scrolledtext
3
4 window = Tk()
5 window.title("Algoritmos")
6 window.geometry('350x200')
7
8 txt = scrolledtext.ScrolledText(window,width=10,height=5)
9 txt.place(relx=0.5, rely=0.5, anchor=CENTER)
10
11 def clicked():
12     print(txt.get(1.2, END))
13
14 btn = Button(window, text="Clique!", command=clicked)
15 btn.grid(column=0, row=1)
16
17 window.mainloop()
```

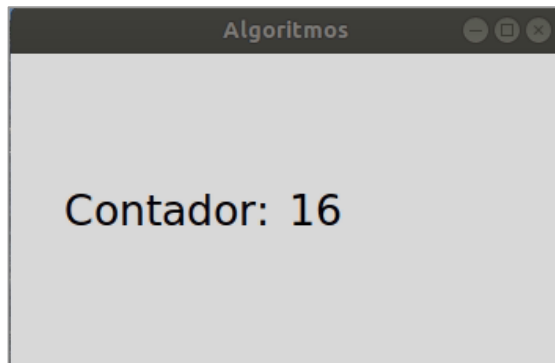
cdefghijklmnopqrstuvxz

GUI - Barra de Menu (*Menu*)



```
1 from tkinter import *
2 from tkinter import Menu
3
4 window = Tk()
5 window.title("Algoritmos")
6 # cria o objeto menu na janela window
7 menu = Menu(window)
8
9 def click():
10     new_window = Tk()
11     new_window.title("Nova janela")
12     new_window.geometry('150x150')
13     label = Label(new_window, text="Janela nova!")
14     label.grid(column=0, row=0)
15     new_window.mainloop()
16
17 # cria menu com nome new_item
18 new_item = Menu(menu)
19
20
21
22 #cria um item do menu
23 menu.add_cascade(label='File', menu=new_item)
24
25 #cria subitems do item
26 new_item.add_command(label='New', command=click)
27 new_item.add_separator()
28 new_item.add_command(label='Edit')
29
30 window.config(menu=menu)
31 window.mainloop()
```

GUI - Atualização da Janela (*after*)



```
1 from tkinter import *
2
3 window = Tk()
4 window.title("Algoritmos")
5 window.geometry('350x200')
6
7 rotulo = Label(window, text="Contador: ", font=("Arial Bold", 20))
8 rotulo.place(relx=0.5,y=100, anchor=E)
9
10 rotulo2 = Label(window, text="", font=("Arial Bold", 20))
11 rotulo2.place(relx=0.5,y=100, anchor=W)
12
13 k = 0
14
15 def loop_function():
16     global k
17     k += 1
18     rotulo2['text'] = k
19     window.after(1000, loop_function)
20
21 loop_function()
22 window.mainloop()
```

Tempo em milissegundos

Conclusão

Python é legal!

- Python é a linguagem do momento:
 - Ciência de dados
 - Inteligência Artificial
 - Machine Learning
- É potente, compacta, legível e uma ótima escolha para desenvolvimento de software.
- Permite o uso de pacotes escritos em C/C++, mantendo o desempenho.

O que faltou?

- Python + C
- Python Científico
- Python para data Science
- E Python para IA.

Fim



“The machine learning algorithm wants to know if we’d like a dozen wireless mice to feed the Python book we just bought.”

Exercícios

Exercício 1: Faça a seguinte interface gráfica para somar dois números

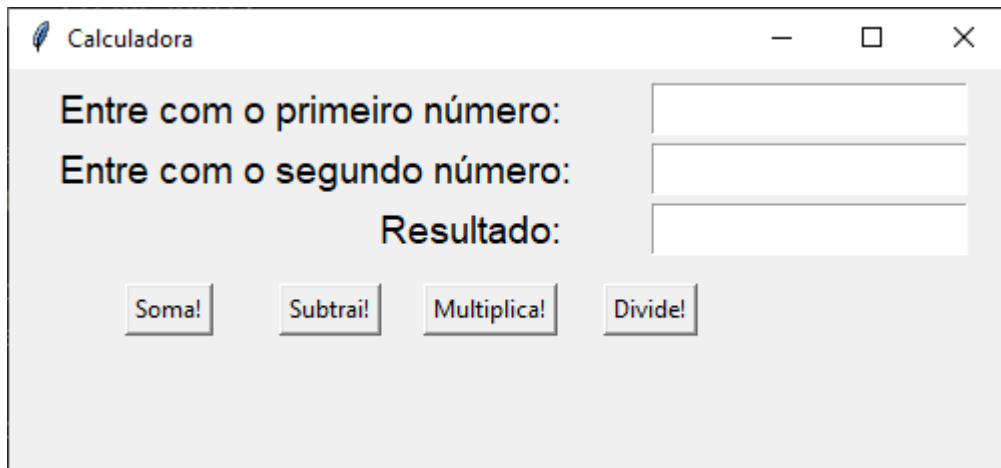
The image shows a window titled "Soma" with a standard Windows-style title bar (minimize, maximize, close buttons). Inside the window, there are three labels on the left: "Entre o primeiro número:", "Entre o segundo número:", and "Resultado:". To the right of each label is a white rectangular input field. Below the "Resultado:" label is a button labeled "Soma!".

Posição dentro da caixa de texto... da posição inicial (0) até o fim da caixa (END)

Para apagar um texto de uma caixa de texto, utilize: **<nome_da_caixa>.delete(0,END)**

Exercício 2

Faça uma calculadora que realize as operações de soma, subtração, multiplicação e divisão.



Calculadora

Entre com o primeiro número:

Entre com o segundo número:

Resultado:

Soma! Subtrai! Multiplica! Divide!

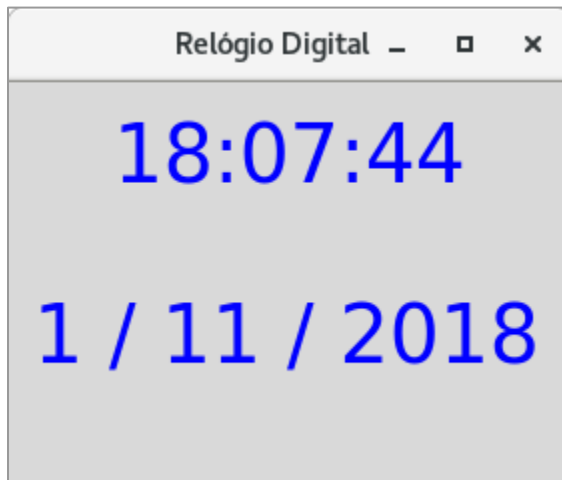
Posição dentro da caixa de texto... da posição inicial (0) até o fim da caixa (END)

Para apagar um texto de uma caixa de texto, utilize: **<nome_da_caixa>.delete(0,END)**

Exercício 3

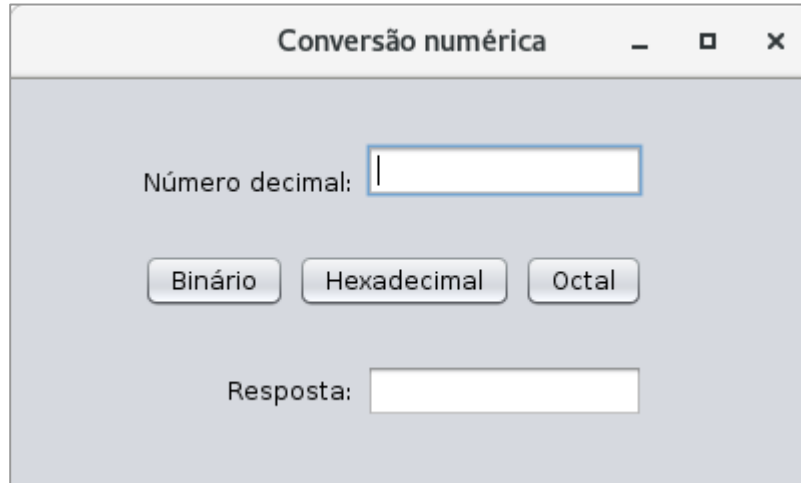
Crie um programa que lê uma letra do alfabeto por uma caixa de texto. Se o usuário digitar **a, e, i, o** ou **u**, seu programa deverá exibir uma mensagem indicando que a letra inserida é uma **vogal** (utilize *caixas de mensagem - messagebox*). Se o usuário digitar **y**, seu programa deve exibir uma mensagem indicando que às vezes **y** é uma **vogal** (depende da língua, no inglês, por exemplo), e às vezes **y** é uma **consoante**. Caso contrário, seu programa deve exibir uma mensagem indicando que o letra é uma **consoante**

Exercício 4: Construa um relógio digital que atualize a cada 1 segundo e exiba, além da hora, o dia, mês e ano. Conforme exemplo:



Utilize o pacote *datetime*

Exercício 5: Faça uma calculadora para transformar números decimais em binários, hexadecimais ou octais. Cada base numérica deve ter um botão para realizar a conversão.



Conversão numérica

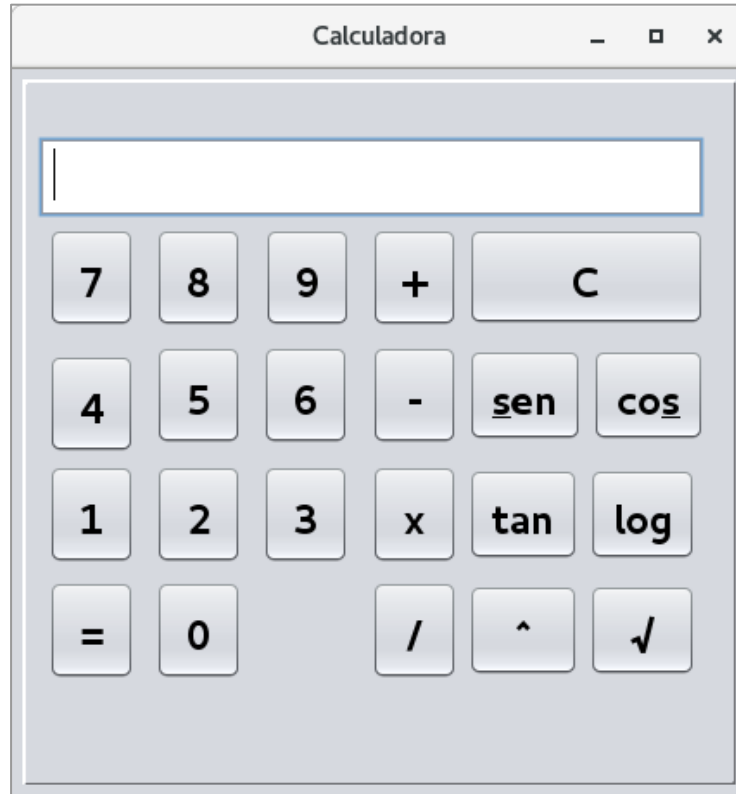
Número decimal:

Resposta:

Use as funções: `hex()`, `bin()` e `oct()`, para converter de decimal para a base desejada.

Exemplo: `num = hex(25)`

Exercício 6: Faça uma calculadora com as 4 operações básicas, potência, sen, cos, tan, log e raiz quadrada.



Exercício 7: Uma **locadora de veículos** te contratou para fazer o aplicativo que controla os aluguéis. Assim, faça uma interface gráfica para cadastro de automóveis (o cadastro deve ser armazenado em arquivo texto). A janela principal deve ter entrada para os seguintes dados:

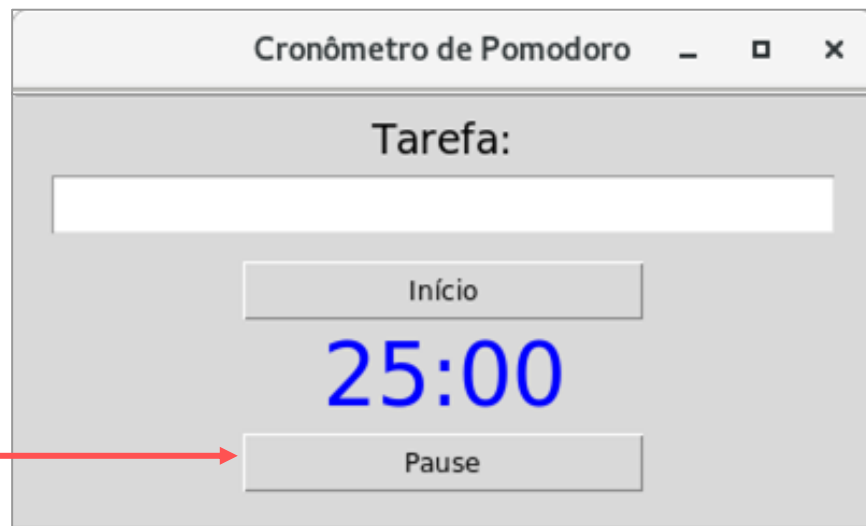
- Marca do veículo
- Modelo
- Ano de fabricação
- Placa
- Km

Utilize **combobox** para os acessórios, onde cada combobox tem como opções ***sim*** ou ***não***:

- Ar condicionado
- Direção hidráulica
- Airbag

Salve um veículo por arquivo texto.

Exercício 8: A **técnica de pomodoro** envolve concentrar-se em uma tarefa por períodos de 25 minutos e, então, parar por um intervalo de tempo curto. Assim, vamos construir um cronômetro que fará a contagem regressiva por 25 minutos (*para testar, vamos construir um que trabalhe com 25 segundos*) e alertará o usuário quando o tempo acabar (*caixa de mensagem*). O alerta deve incluir o nome da tarefa (que foi digitado na caixa de entrada). A janela segue o *layout* abaixo:



The image shows a window titled "Cronômetro de Pomodoro". Inside the window, there is a label "Tarefa:" followed by a text input field. Below the input field is a button labeled "Início". In the center of the window, the time "25:00" is displayed in large blue digits. Below the time is a button labeled "Pause".

O botão de *pause* deve pausar a contagem ou voltar a contar se já estiver pausada