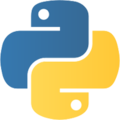
**+**

**RAPPORT DE PROJET :**

**Projet : CREER UNE CALCULATRICE AVEC LA BIBLIOTHEQUE TKINTER DE PYTHON AVEC LES FONCTIONNALITés TEL QUE : ( + *, -, x, /,* √*, x^2 et exp^x )***

****

**Membre du groupe : Enseignant :**

***BAGA JOSEPH Mr :***

***YAMEOGO MICAEL Kyelem Yacouba***

**Introduction**

Dans le cadre de notre cours d'initiation à la programmation en Python, nous avons reçu comme devoir de réaliser un projet pratique pour mettre en application les concepts et techniques étudiés en classe. L'objectif de ce projet était de concevoir et de développer une calculatrice graphique en utilisant les bibliothèques Python tkinter pour l'interface utilisateur et PIL (Python Imaging Library) pour la manipulation des images.

L'ajout de la bibliothèque PIL a été une initiative personnelle pour rendre l'interface plus attrayante et moderne. Ce projet vise à évaluer notre compréhension des bases de la programmation en Python, notamment la création d'interfaces graphiques, la gestion des événements, et l'implémentation de fonctionnalités mathématiques. La réalisation de ce projet permet également de développer des compétences en matière de conception logicielle, de gestion des erreurs, et d'amélioration de l'ergonomie des applications.

**Description du Projet**

Ce projet consiste à développer une calculatrice graphique qui peut effectuer des opérations de base : addition, soustraction, multiplication et division, ainsi que des opérations avancées telles que la racine carrée, l'exponentiation et la mise au carré. L'interface utilisateur comprendra des boutons pour chaque chiffre (0-9), chaque opération, ainsi qu'un affichage pour montrer les résultats.

**Aperçu sur Tkinter**

Tkinter est la bibliothèque standard de Python pour la création d'interfaces graphiques. Elle fournit un moyen simple et efficace de concevoir des applications avec des fenêtres, des boutons, des champs de texte, et d'autres éléments d'interface utilisateur. Dans ce projet, Tkinter sera utilisé pour créer l'interface graphique de la calculatrice, permettant ainsi une interaction utilisateur intuitive et visuellement agréable. La bibliothèque **PIL** (Python Imaging Library) est aussi associé pour créer des images avec des coins arrondis.

**Guide des sections**

1. *Importation des Bibliothèques*

import tkinter as tk

from PIL import Image, ImageDraw, ImageTk

import math

L’ importation de la bibliothèque tkinter ici permet de crée une interface graphique et la bibliothèque math Fournit des fonctions mathématiques pour évaluer les expressions. IL y a aussi la bibliothèque PIL utilisé pour créer des images avec des coins arrondis.

**Le code est structuré en plusieurs fonctions distinctes, chacune ayant un rôle spécifique :**

1. **creer\_rectangle\_arrondi(largeur, hauteur, rayon, couleur)** : Crée une image de rectangle avec des coins arrondis et une couleur de fond spécifiée.

# Fonction pour créer une image avec des coins arrondis

def creer\_rectangle\_arrondi(largeur, hauteur, rayon, couleur):

image = Image.new("RGBA", (largeur, hauteur), (0, 0, 0, 0))

draw = ImageDraw.Draw(image)

draw.rounded\_rectangle((0, 0, largeur, hauteur), radius=rayon, fill=couleur)

return image

1. *Évaluation des expressions mathématique*

# Fonction pour évaluer l'expression

def evaluer(expression):

try:

if '√' in expression:

parts = expression.split('√')

if parts[0] == '':

result = math.sqrt(float(parts[1]))

else:

result = float(parts[0]) \* math.sqrt(float(parts[1]))

elif 'e^' in expression:

parts = expression.split('e^')

if parts[0] == '':

result = math.exp(float(parts[1]))

else:

result = float(parts[0]) \* math.exp(float(parts[1]))

else:

result = eval(expression.replace('×', '\*').replace('÷', '/').replace('−', '-').replace('^', '\*\*').replace('x²', '\*\*2'))

return f"= {result}"

except Exception as e:

return "Erreur"

La fonction evaluer(expression) évalue une expression mathématique donnée sous forme de chaîne de caractères. Elle commence par vérifier si l'expression contient une racine carrée ('√') ou une exponentielle ('e^'), et traite ces cas spécifiques en utilisant math.sqrt et math.exp. Pour les autres expressions, elle utilise eval après avoir remplacé certains symboles (×, ÷, −, ^, x²) par leurs équivalents Python (\*, /, -, \*\*, \*\*2). Si l'évaluation réussit, le résultat est retourné sous la forme d'une chaîne de caractères précédée de '='. En cas d'erreur (valeur, syntaxe, ou autre), des messages d'erreur spécifiques sont retournés.

3)*Affichage de la fenêtre de la calculatrice*

# Fonction pour afficher l'image avec tkinter

def afficher\_calculatrice():

root = tk.Tk()

root.title("Calculatrice")

root.configure(bg='#8E7ECE')

screen\_width = root.winfo\_screenwidth()

screen\_height = root.winfo\_screenheight()

main\_width = 1000

main\_height = 700

main\_radius = 15

main\_color = "#8E7ECE"

x = (screen\_width - main\_width) // 2

y = (screen\_height - main\_height) // 2

root.geometry(f"{main\_width}x{main\_height}+{x}+{y}")

main\_image = creer\_rectangle\_arrondi(main\_width, main\_height, main\_radius, main\_color)

main\_tk\_image = ImageTk.PhotoImage(main\_image)

main\_label = tk.Label(root, image=main\_tk\_image, bg='#8E7ECE')

main\_label.image = main\_tk\_image

main\_label.pack(padx=20, pady=20)

calc\_width = 800

calc\_height = 600

calc\_radius = 15

calc\_color = "#2D2A37"

calc\_image = creer\_rectangle\_arrondi(calc\_width, calc\_height, calc\_radius, calc\_color)

calc\_tk\_image = ImageTk.PhotoImage(calc\_image)

calc\_label = tk.Label(main\_label, image=calc\_tk\_image, bg='#8E7ECE')

calc\_label.image = calc\_tk\_image

calc\_label.place(relx=0.5, rely=0.5, anchor="center", width=calc\_width, height=calc\_height)

creer\_interface\_calculatrice(calc\_label)

root.mainloop()

La fonction afficher\_calculatrice() initialise et affiche la fenêtre principale de l'application de calculatrice. Elle crée une instance de Tk et configure le titre et la couleur de fond. Elle centre la fenêtre sur l'écran en calculant les coordonnées x et y. Ensuite, elle crée une image avec des coins arrondis pour la fenêtre principale et l'affiche dans un Label. De la même manière, elle crée et affiche une image pour la calculatrice elle-même. La fonction creer\_interface\_calculatrice(calc\_label) est appelée pour ajouter les boutons et le champ d'affichage de la calculatrice. Enfin, root.mainloop() lance la boucle principale de l'interface graphique.

*4. Création de l'interface utilisateur de la calculatrice*

# Fonction pour créer l'interface de la calculatrice

def creer\_interface\_calculatrice(parent):

buttons = [

['x^y', '(', ')', 'DEL'],

['e^', 'x²', '√', '÷'],

['7', '8', '9', '×'],

['4', '5', '6', '−'],

['1', '2', '3', '+'],

['.', '0', '⌫', '=' ]

]

display = tk.Entry(parent, font=("Arial", 24), bg="#2D2A37", fg="white", justify="right", bd=0)

display.grid(row=0, column=0, columnspan=4, padx=10, pady=10, sticky="nsew")

parent.display = display

for i, row in enumerate(buttons):

for j, button in enumerate(row):

btn = tk.Button(parent, text=button, font=("Arial", 18), bg="#3B3A4F" if button.isdigit() else "#5E5A8B", fg="white", bd=0, relief="flat")

btn.grid(row=i+1, column=j, padx=10, pady=10, sticky="nsew")

btn.config(command=lambda b=button: clic\_bouton(parent, b))

for i in range(len(buttons)):

parent.rowconfigure(i+1, weight=1)

for j in range(4):

parent.columnconfigure(j, weight=1)

La fonction creer\_interface\_calculatrice(parent) configure l'interface utilisateur de la calculatrice. Elle crée un champ d'affichage (Entry) en haut de la calculatrice, avec une police spécifique et des couleurs de fond et de texte, aligné à droite. Chaque bouton de la calculatrice est ensuite créé en parcourant la liste BUTTONS. La couleur du bouton est déterminée par le type (chiffre ou opérateur). Les boutons sont placés dans une grille (grid), avec des commandes qui appellent clic\_bouton lorsqu'ils sont cliqués. Enfin, les lignes et colonnes de la grille sont configurées pour s'étendre proportionnellement (weight=1), assurant ainsi une mise en page flexible et réactive.

*5. Gestion des clics sur les boutons*

# Fonction appelée lors du clic sur un bouton

def clic\_bouton(parent, button):

display = parent.display

current\_text = display.get()

if current\_text == "Erreur" and button.isdigit():

display.delete(0, tk.END)

current\_text = ""

if button == '⌫':

display.delete(len(current\_text) - 1, tk.END)

elif button == 'DEL':

display.delete(0, tk.END)

elif button == '=':

result = evaluer(current\_text)

display.delete(0, tk.END)

display.insert(tk.END, result)

elif button == 'x^y':

display.insert(tk.END, '^')

elif button == 'e^':

display.insert(tk.END, 'e^')

elif button == '√':

display.insert(tk.END, '√')

elif button == 'x²':

display.insert(tk.END, 'x²')

else:

display.insert(tk.END, button)

La fonction clic\_bouton(parent, button) gère les actions des boutons de la calculatrice. Elle récupère et met à jour le texte affiché. Si le bouton cliqué est '⌫', il supprime le dernier caractère; si c'est 'DEL', il efface tout le texte. Si c'est '=', il évalue l'expression et affiche le résultat. Les boutons spéciaux ('x^y', 'e^', '√', 'x²') sont directement ajoutés à l'affichage, et les autres boutons insèrent leur valeur dans l'affichage actuel.

**Exemples d’opération possible**

1. **Addition :** 2 + 3 = 5
2. **Soustraction :** 5 − 2 = 3
3. **Multiplication :** 4 × 3 = 12
4. **Division :** 8 ÷ 2 = 4
5. **Puissance :** 2^3 = 8
6. **Racine Carrée :** √16 = 4, 3√27 = 3
7. **Exponentielle :** e^2 = 7.38905609893065
8. **Carré :** 4x² = 16
9. **Priorité des Opérations :** (2 + 3) × 4 = 20

**Note : Utilisation et Installation**

**Pré-requis :**

**Python 3.x**

Bibliothèque PIL (Pillow)

Installation de Pillow :

**pip install pillow**

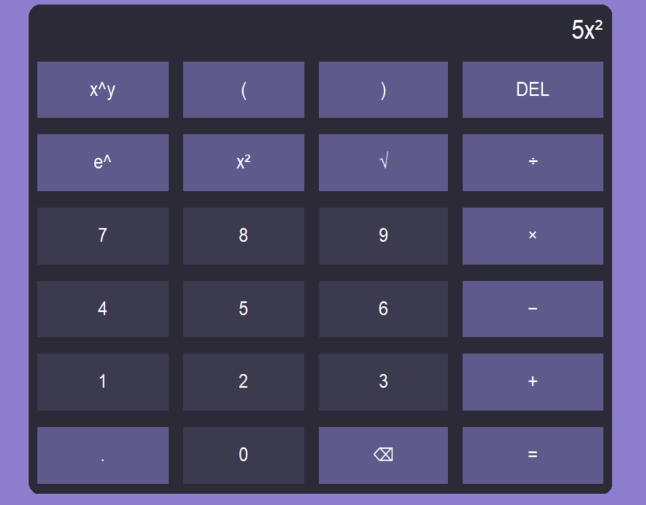
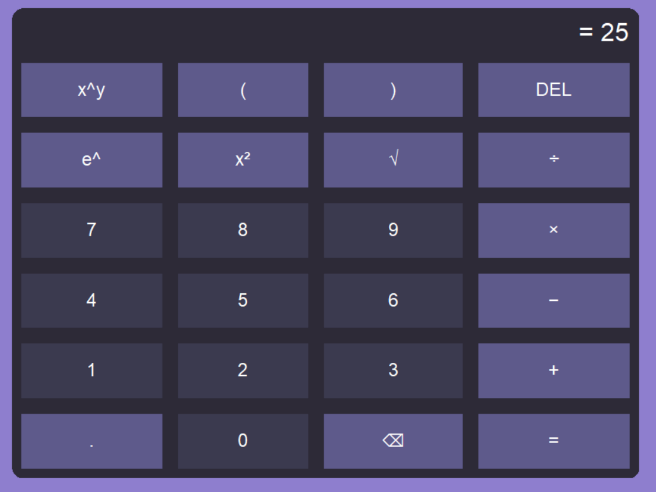
**reference**

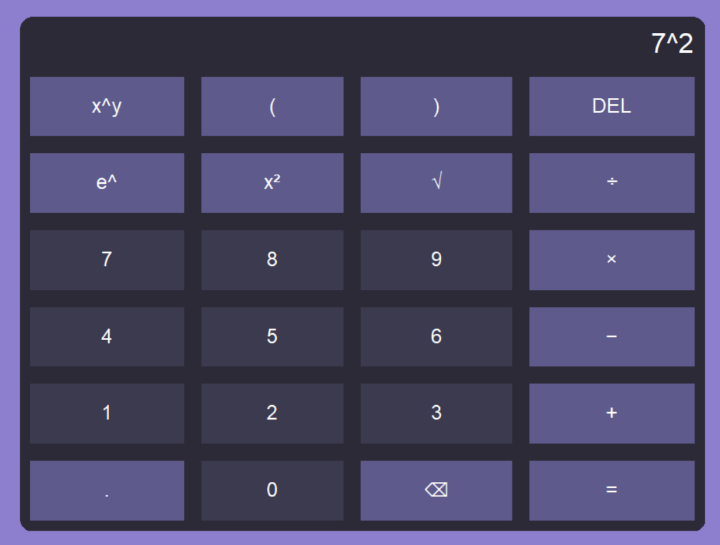
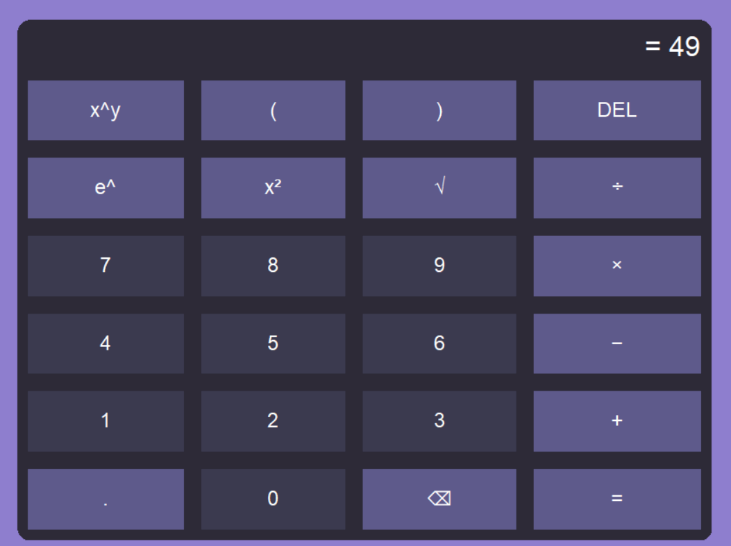
* Documentation officielle de tkinter : <https://docs.python.org/3/library/tkinter.html>
* Documentation officielle de Pillow : <https://pillow.readthedocs.io/en/stable/>
* Documentation officielle math [math — Mathematical functions — Python 3.12.4 documentation](https://docs.python.org/3/library/math.html)

**Conclusion**

Ce code présente une calculatrice graphique complète utilisant Tkinter et PIL. Les opérations mathématiques avancées sont gérées de manière appropriée, et l'interface est configurée pour être intuitive et visuellement agréable

**Capture d’écrans**

****

****