

**SUP MTI**  
École Supérieure de Management,  
de Télécommunication et d'Informatique

**RECONNUE PAR L'ÉTAT**

# **PYTHON**

## **SÉANCE NUM 2**



Enseignant : GHANEM

# Plan de la Séance 2

## Sommaire :

Partie 1 : Tutoriel : mon premier code en python  
(comparaison interprété VS compilé)

Partie 2 : Types de variables en Python

Partie 3 : Structures conditionnelles et Structures itératives










Partie 4 : Fonctions













Partie 5 : l'équivalent des tableaux en Python ?

Partie 6 : l'équivalent des structures en Python ?

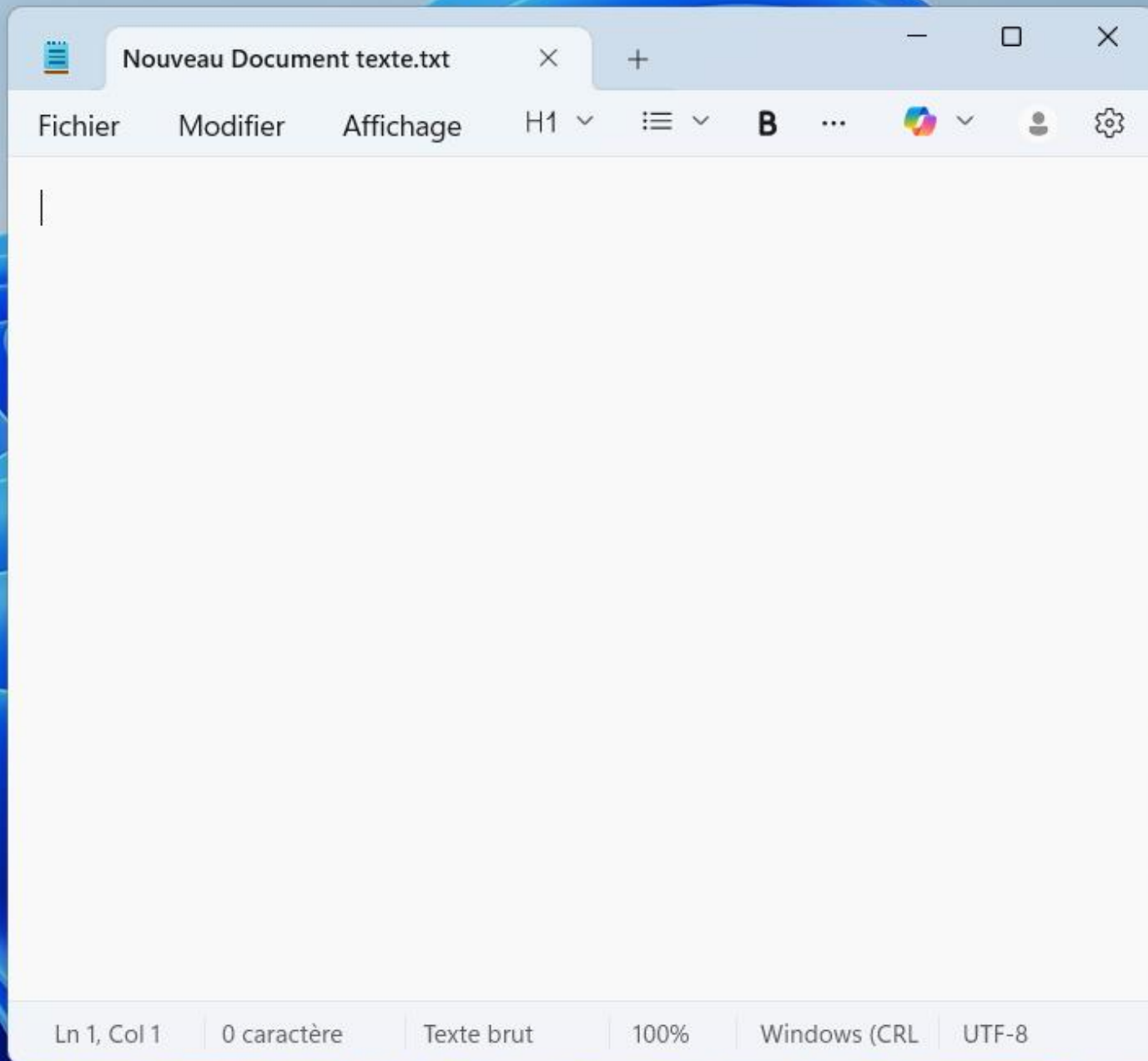
Partie 7 : Définition simple d'une classe et son utilisation

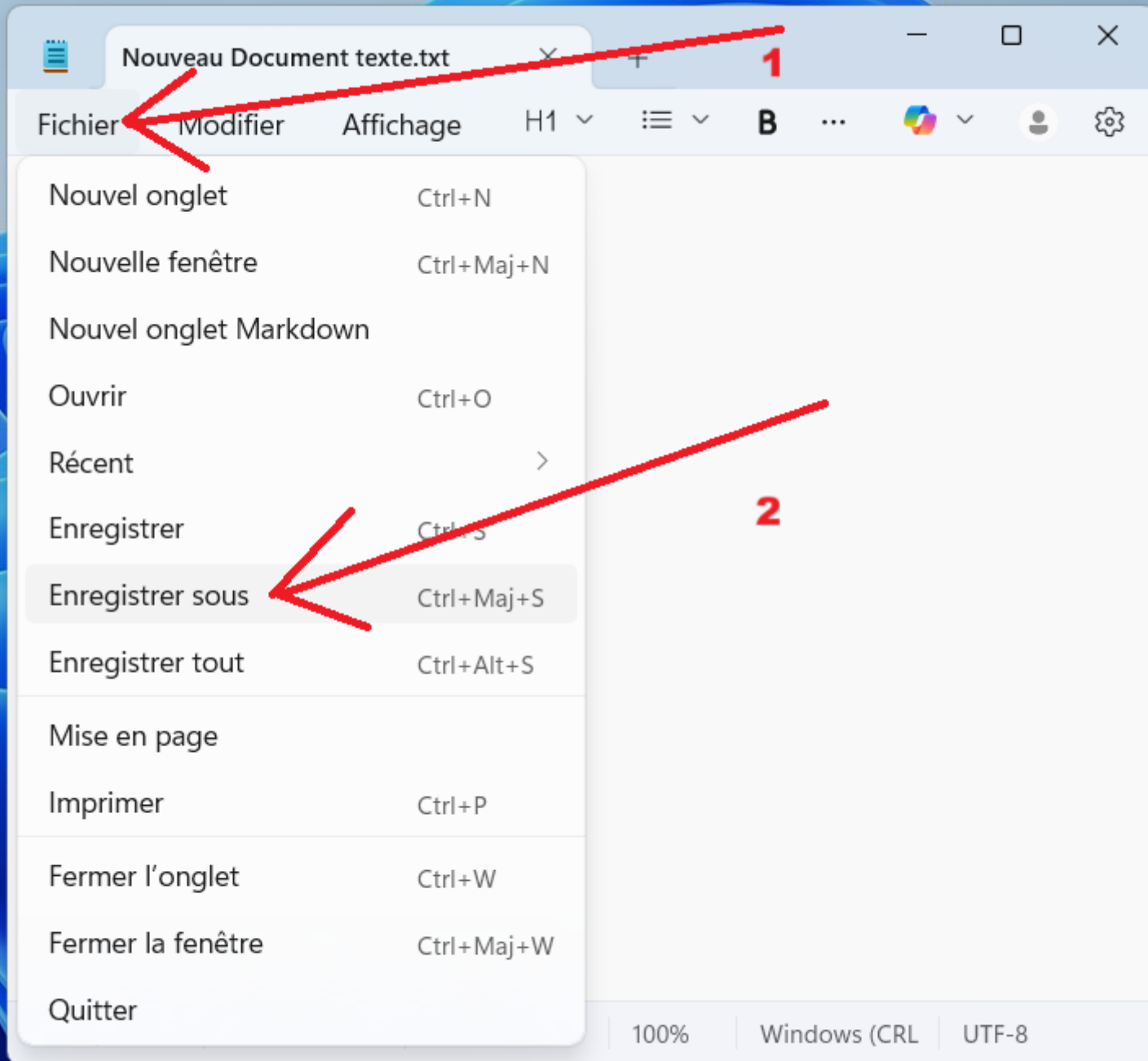
# **Partie 1 : Tutoriel : mon premier code en python (comparaison interprété VS compilé)**

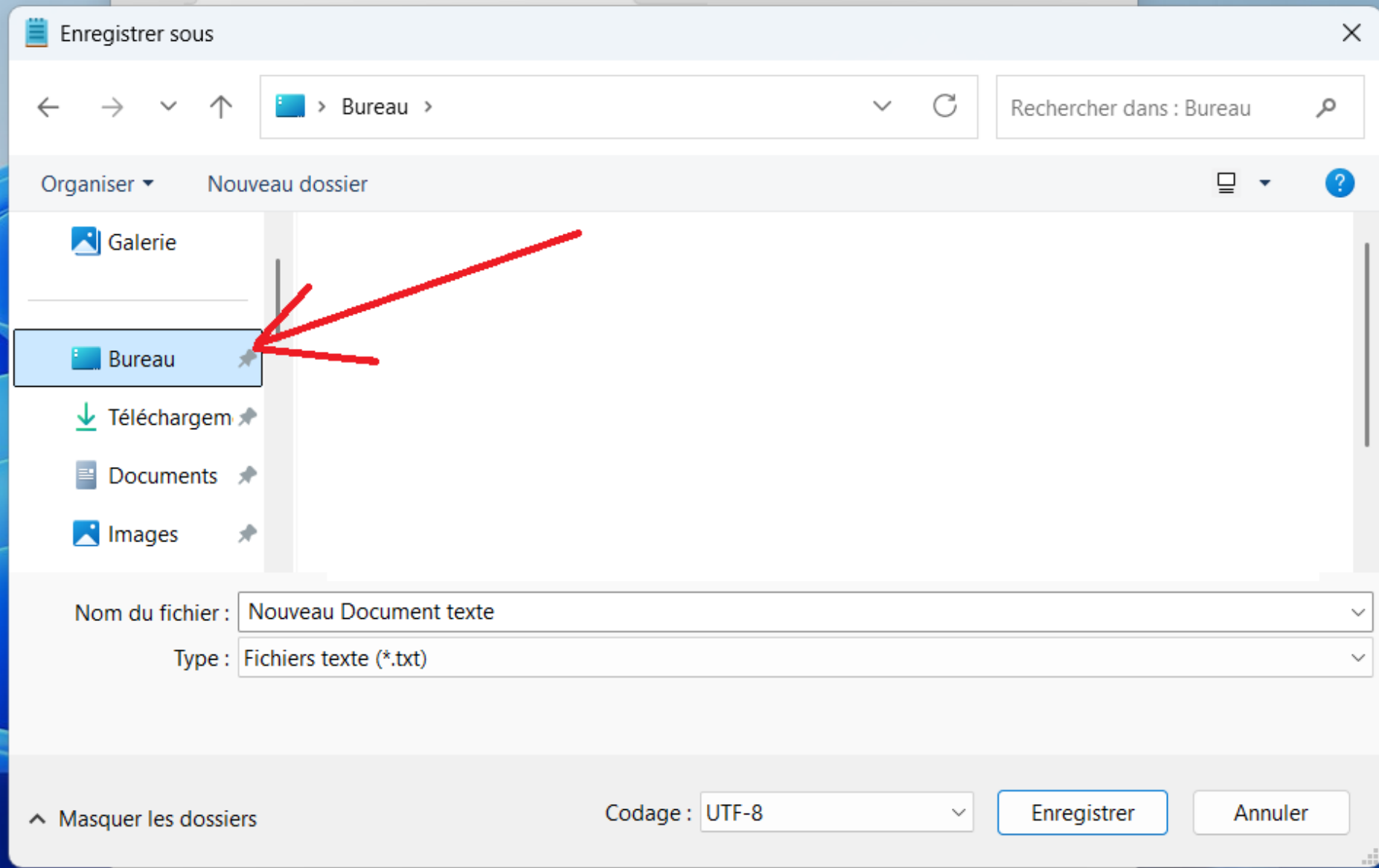
-  Affichage >
-  Trier par >
-  Actualiser
-  Annuler la suppression Ctrl+Z
-  Nouveau >
-  Paramètres d'affichage
-  Personnaliser
-  Ouvrir dans le Terminal
-  Afficher d'autres d'options

-  Dossier
-  Raccourci
-  Microsoft Access Database
-  Image bitmap
-  Document Microsoft Word
-  Microsoft Access Database
-  Présentation Microsoft PowerPoint
-  Microsoft Publisher Document
-  Archive WinRAR
-  Document texte
-  Feuille de calcul Microsoft Excel
-  Archive WinRAR ZIP

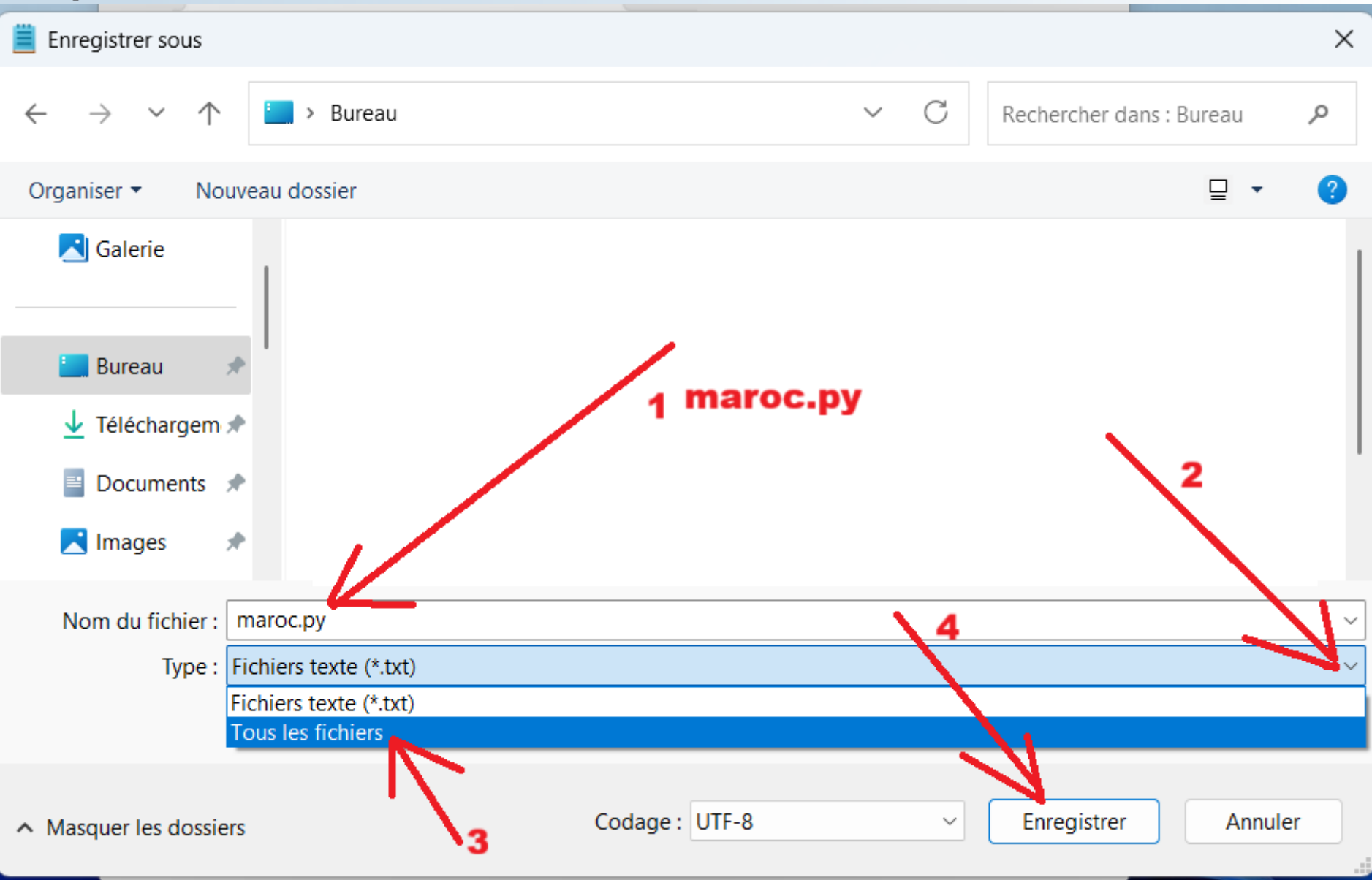


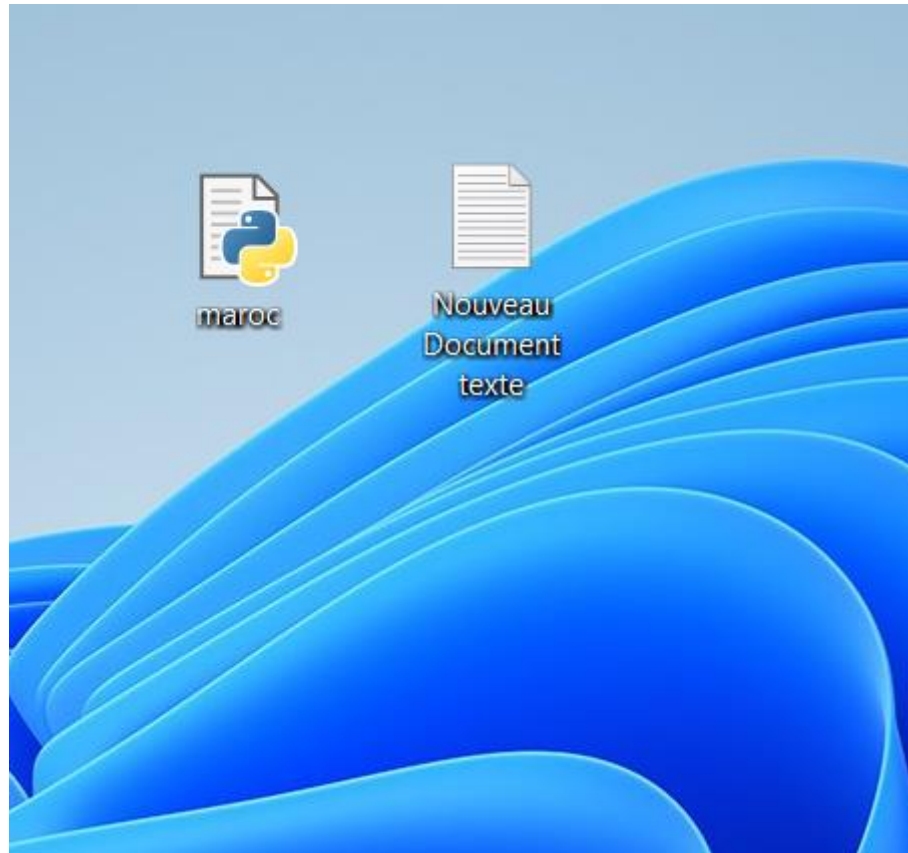


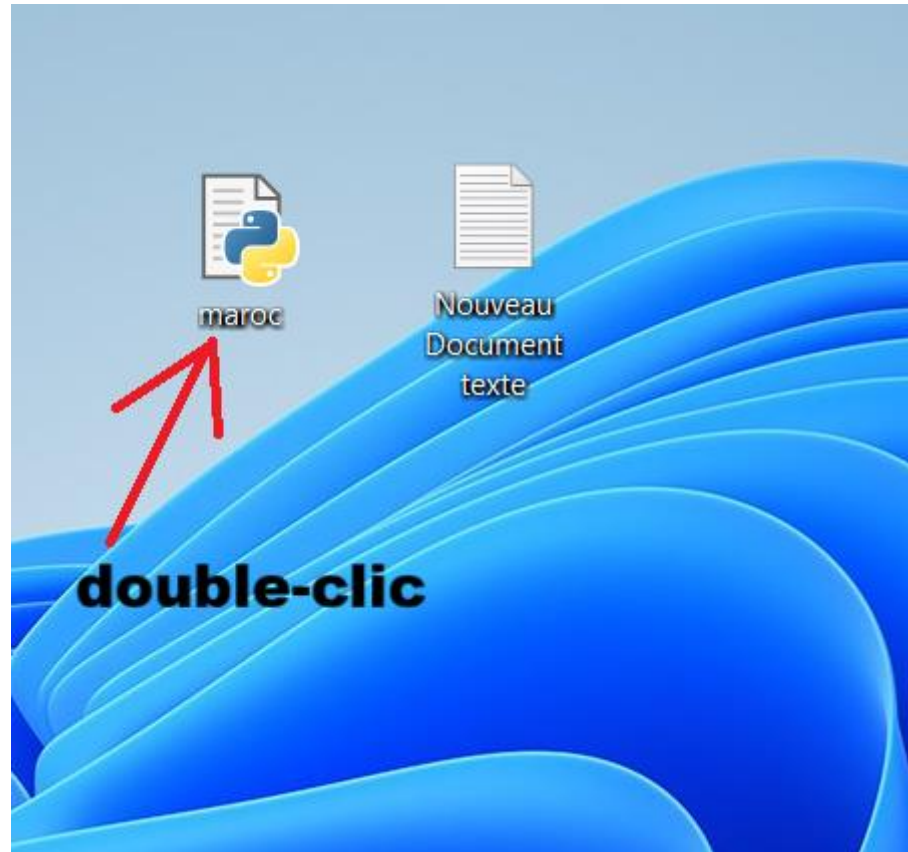






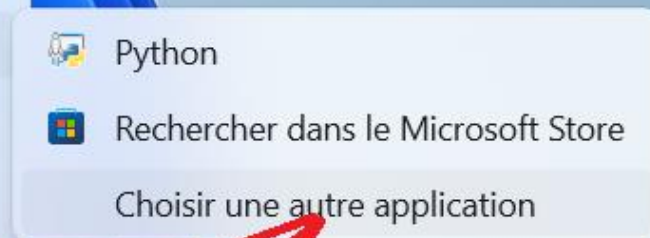
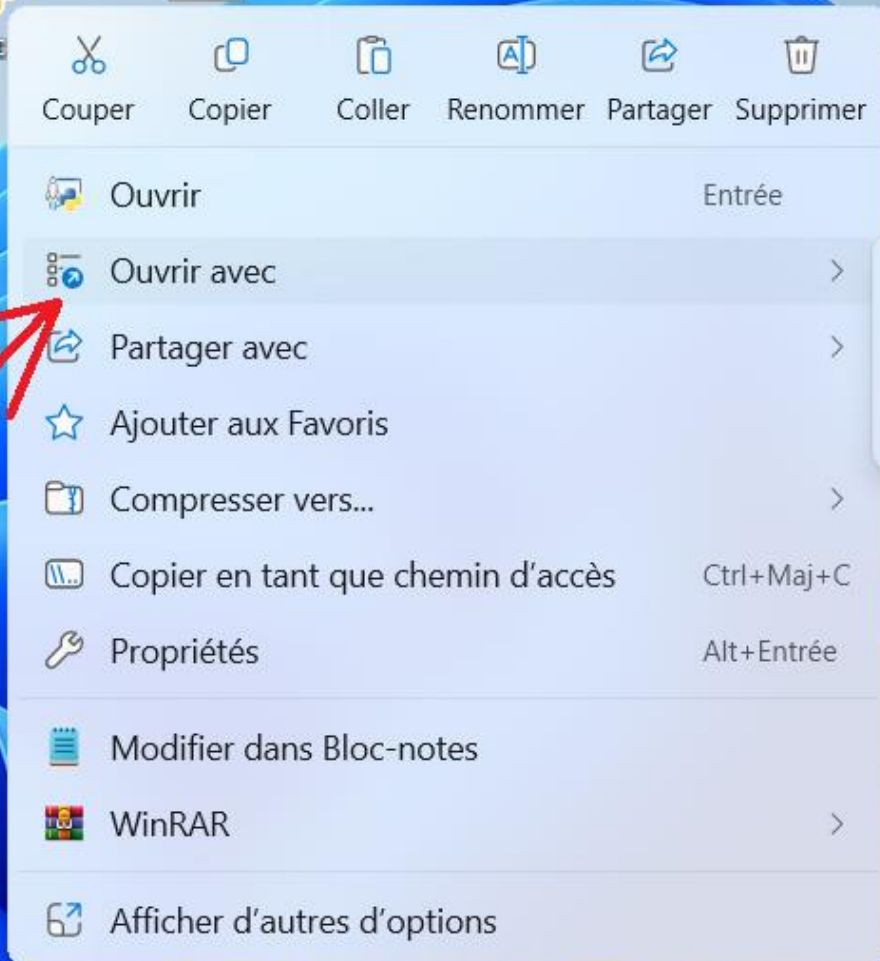






**Malheureusement, l'écran d'exécution apparaît et disparaît rapidement.**

1  
**clic droit**



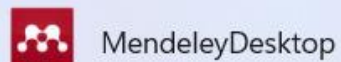
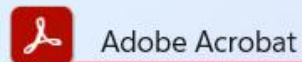
## Sélectionner une application pour ouvrir ce fichier .py

Application par défaut



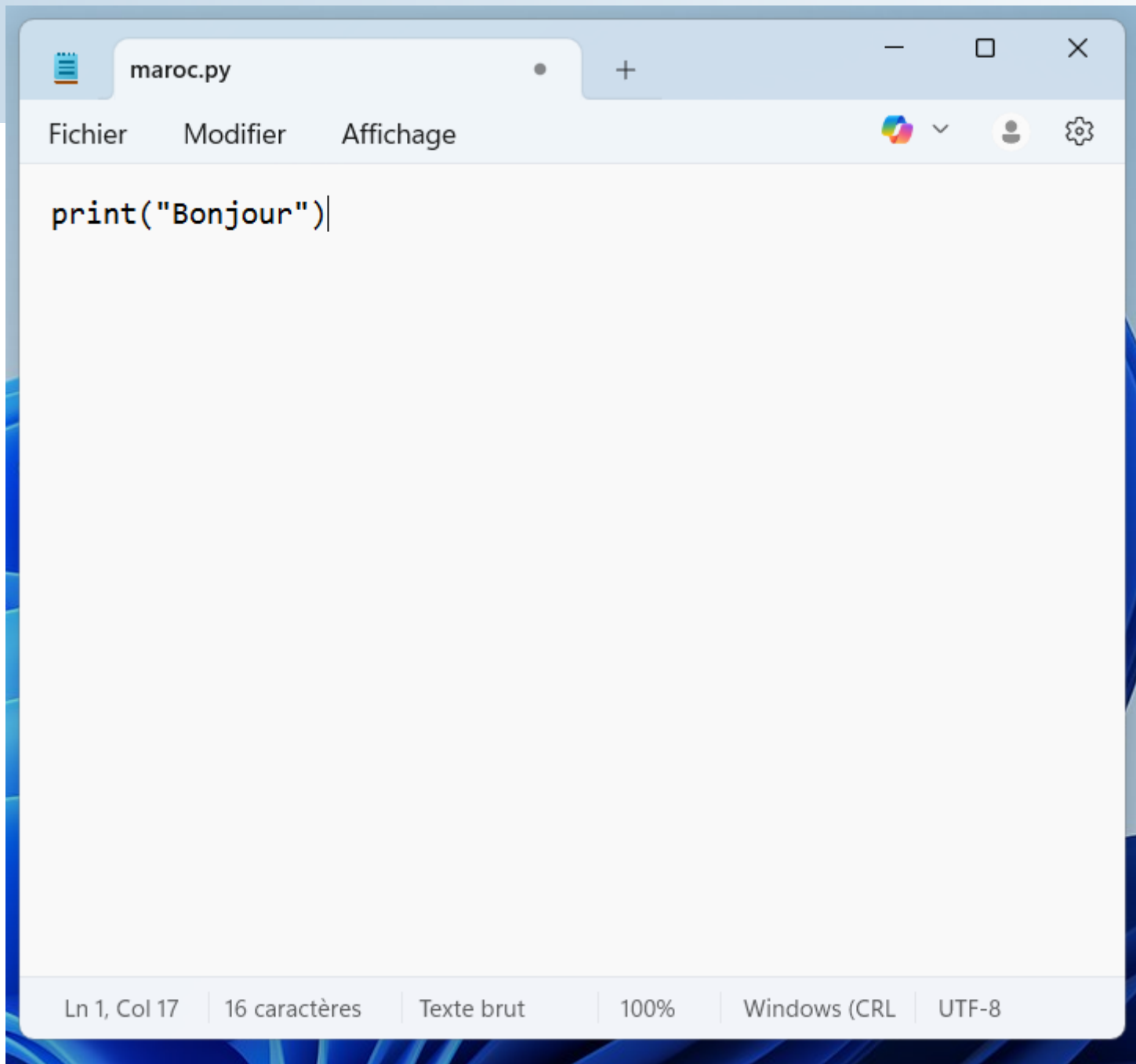
**double-clic**

Autres options



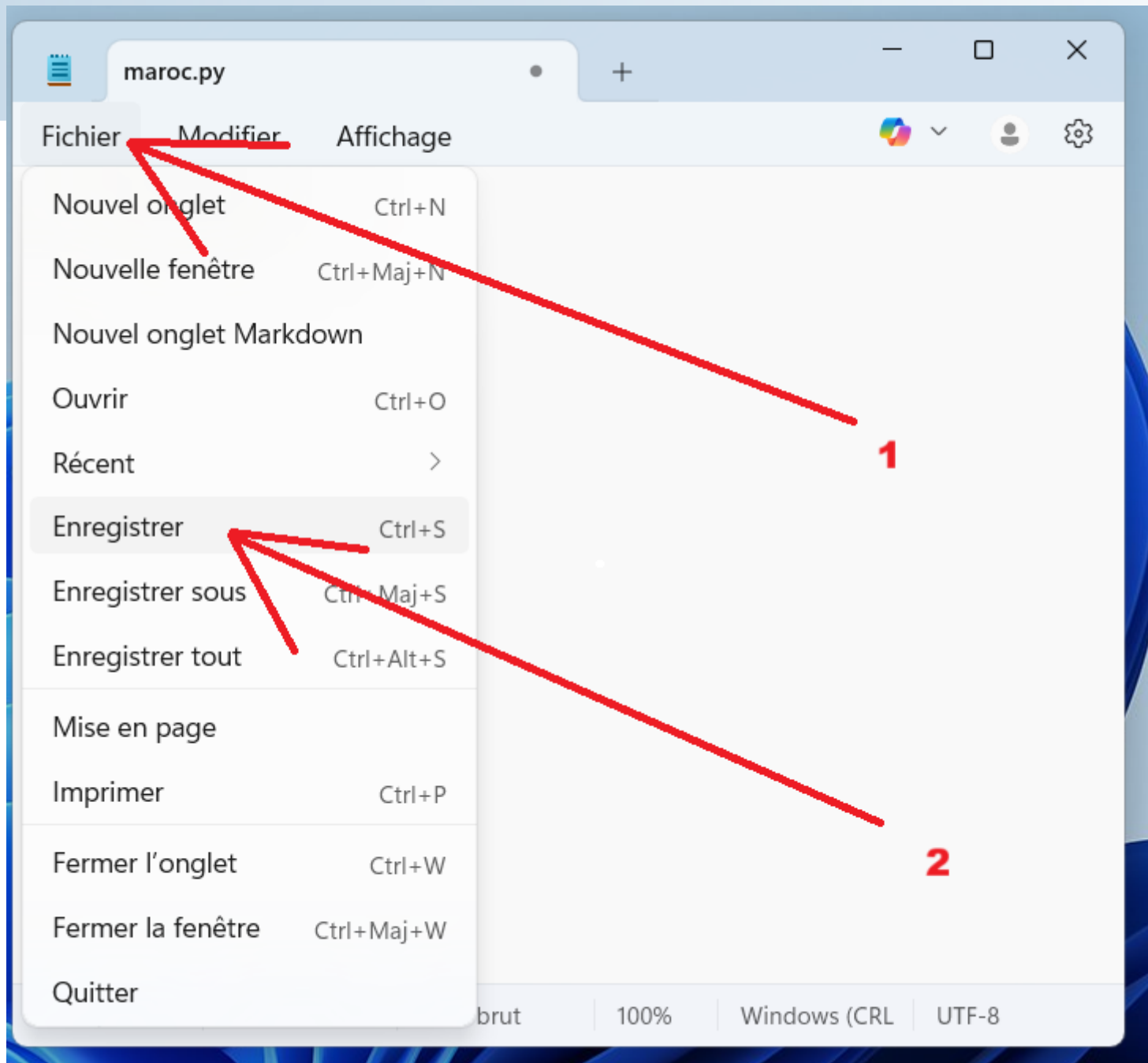
Toujours

Une seule fois

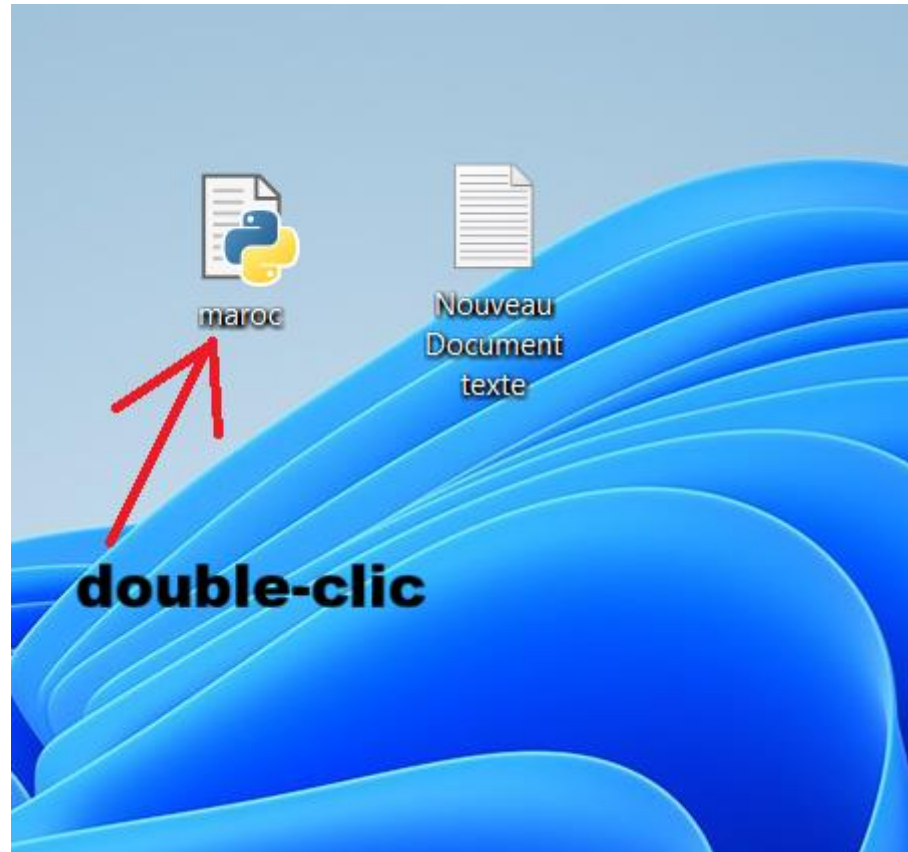


The image shows a web-based code editor interface. At the top, there is a tab labeled "maroc.py". Below the tab, there is a menu bar with three items: "Fichier", "Modifier", and "Affichage". To the right of the menu bar, there are three icons: a colorful cube, a person icon, and a gear icon. The main area of the editor is white and contains the text `print("Bonjour")` with a cursor at the end of the line. At the bottom of the editor, there is a status bar with the following information: "Ln 1, Col 17", "16 caractères", "Texte brut", "100%", "Windows (CRL", and "UTF-8".

```
print("Bonjour")
```





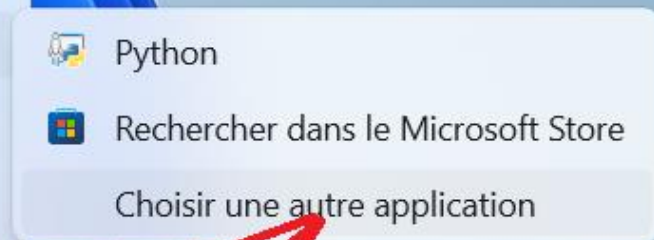
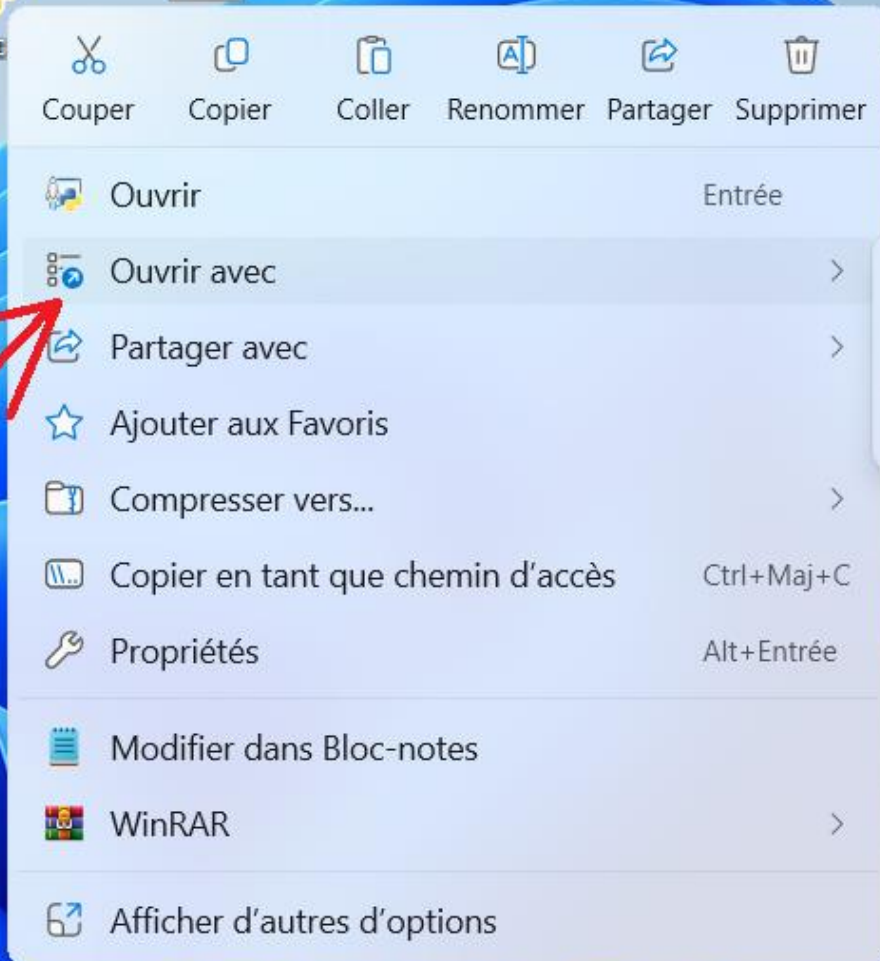


**Encore une fois , l'écran d'exécution apparaît et disparaît rapidement !!!!**

comment bloquer l'écran ?

On va déclarer un entier  $k$  , et on appelle la fonction qui attend la saisie d'un nombre sur le clavier :

1  
**clic droit**



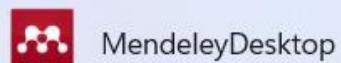
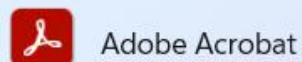
## Sélectionner une application pour ouvrir ce fichier .py

Application par défaut



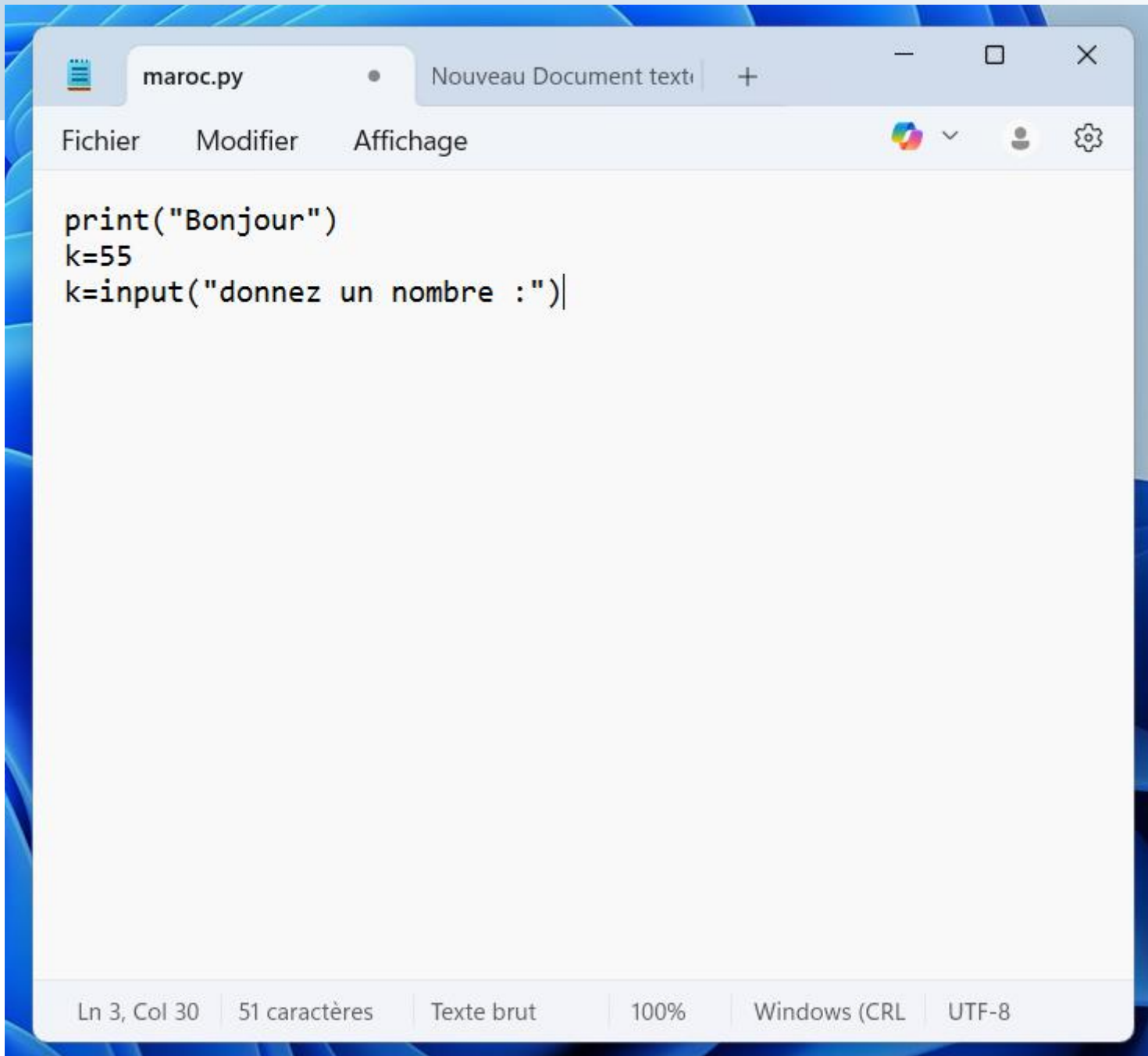
**double-clic**

Autres options



Toujours

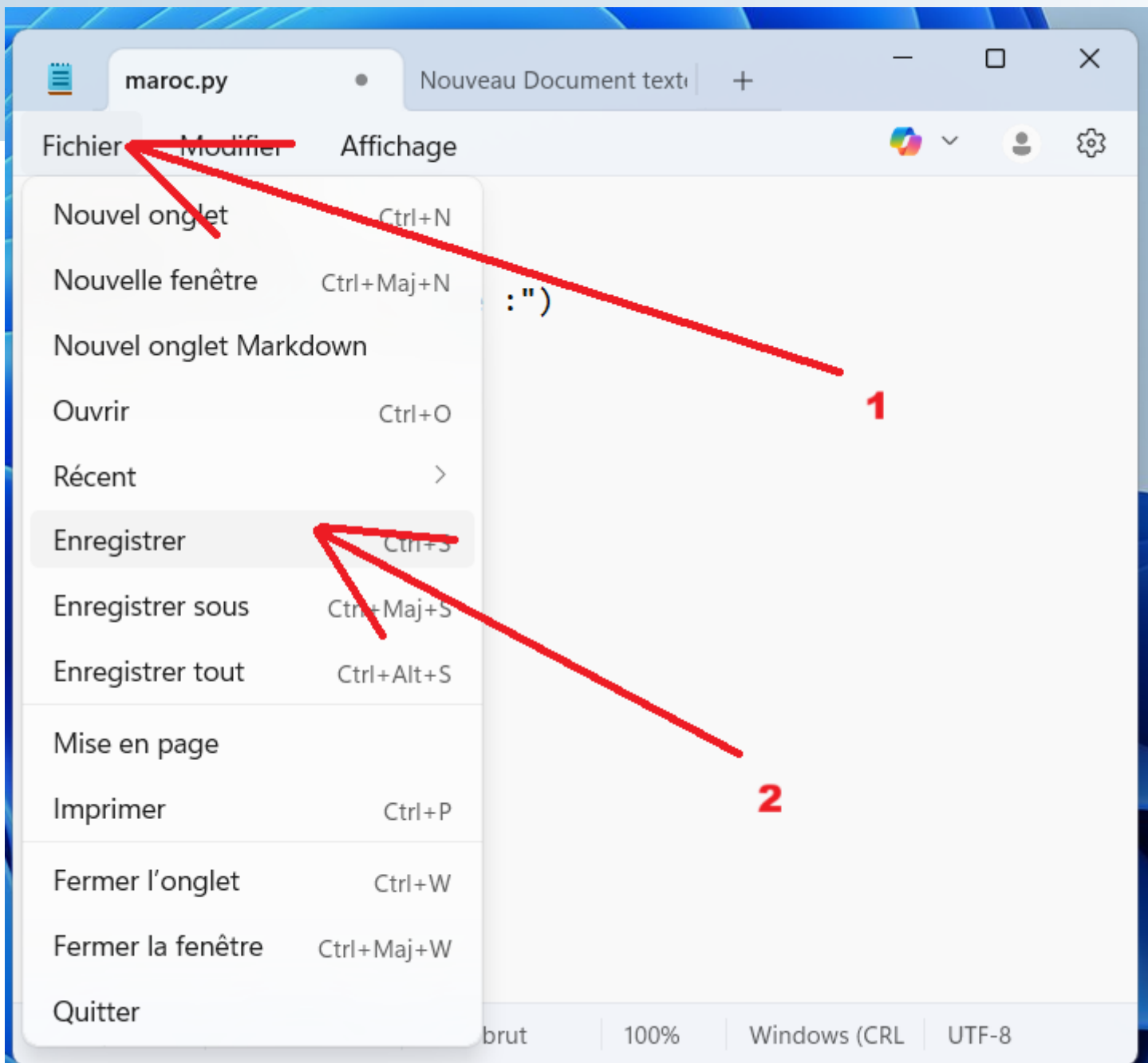
Une seule fois



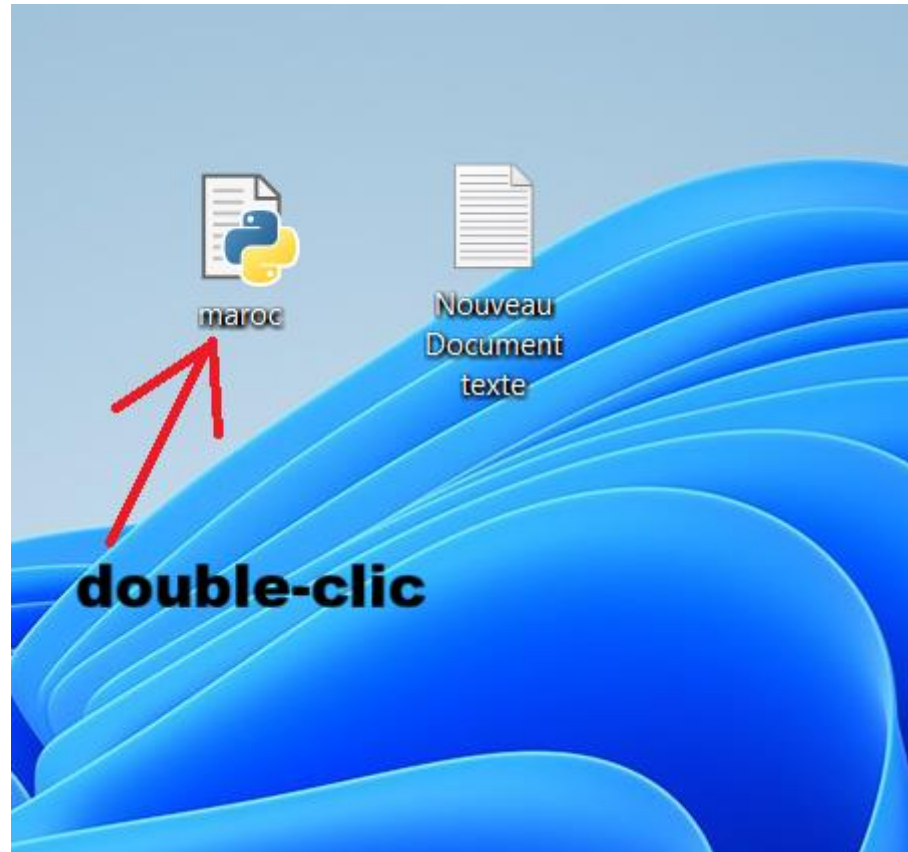
The image shows a screenshot of a text editor window. The window has a title bar with the name 'maroc.py' and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with 'Fichier', 'Modifier', and 'Affichage'. The main editing area contains the following Python code:

```
print("Bonjour")  
k=55  
k=input("donnez un nombre :")|
```

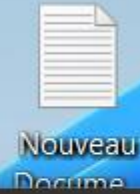
The status bar at the bottom of the window displays the following information: 'Ln 3, Col 30', '51 caractères', 'Texte brut', '100%', 'Windows (CRL', and 'UTF-8'.







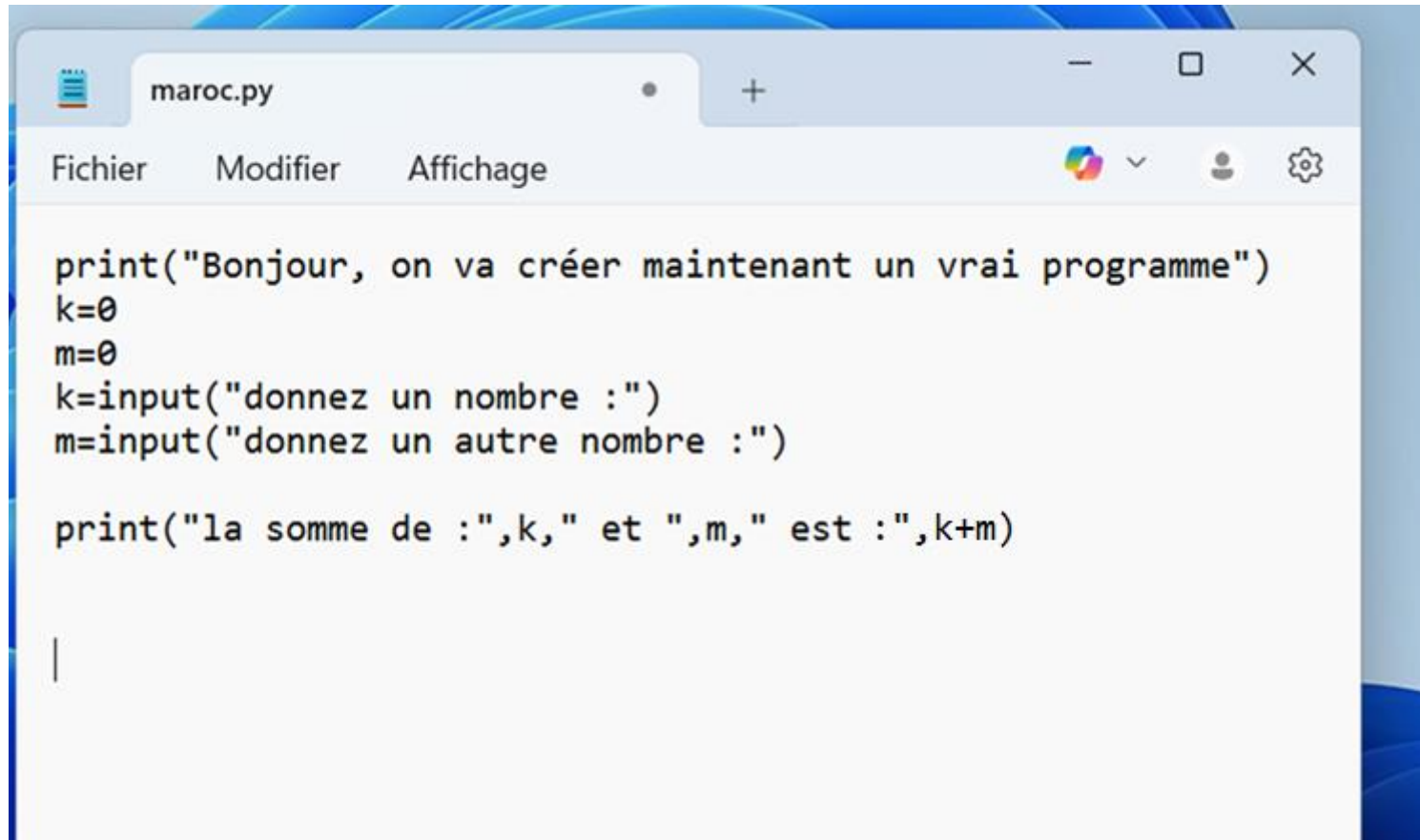




C:\Users\DELL\AppData\Local



Bonjour  
donnez un nombre :

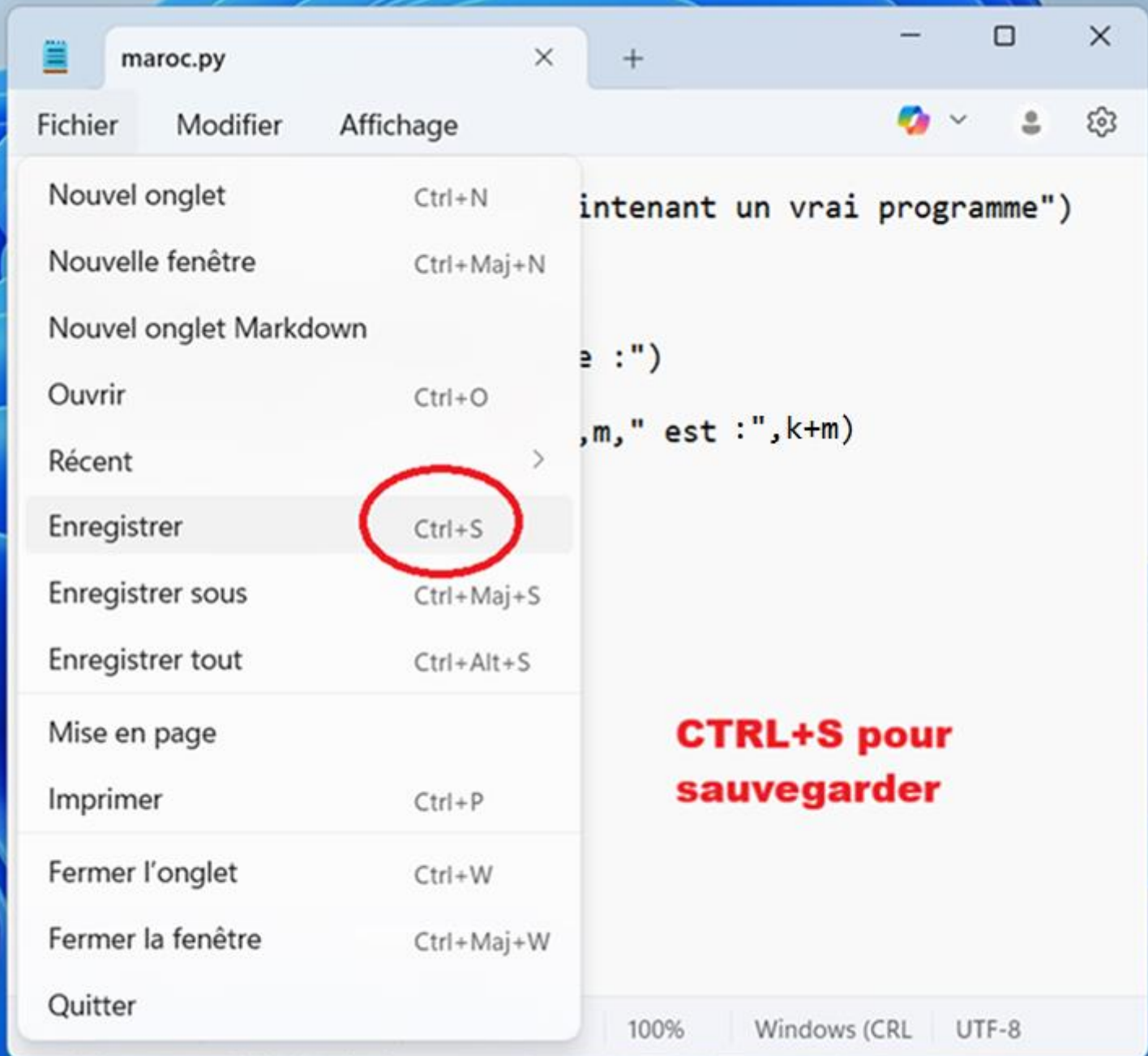


The screenshot shows a web browser window with a single tab titled "maroc.py". The browser's menu bar includes "Fichier", "Modifier", and "Affichage". The main content area displays a Python script with the following code:

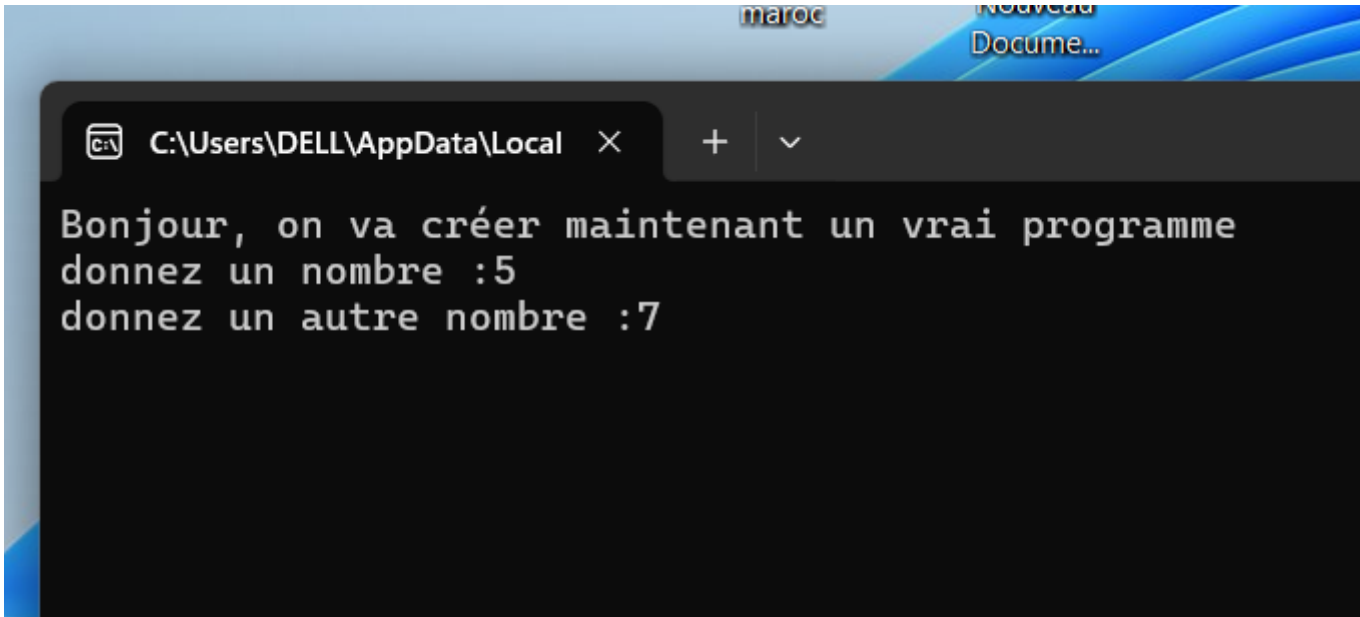
```
print("Bonjour, on va créer maintenant un vrai programme")
k=0
m=0
k=input("donnez un nombre :")
m=input("donnez un autre nombre :")

print("la somme de :",k," et ",m," est :",k+m)

|
```

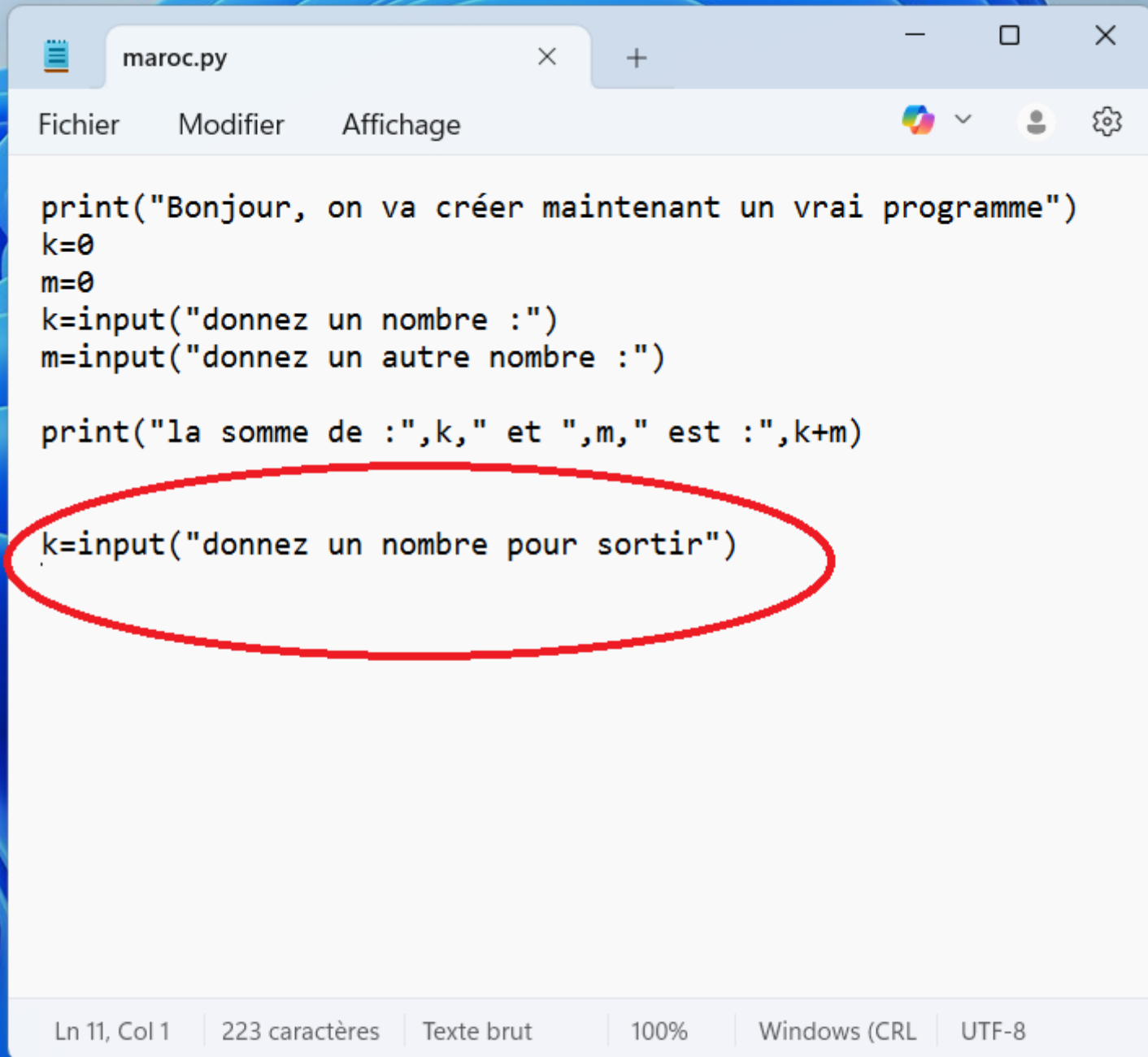


**CTRL+S pour  
sauvegarder**



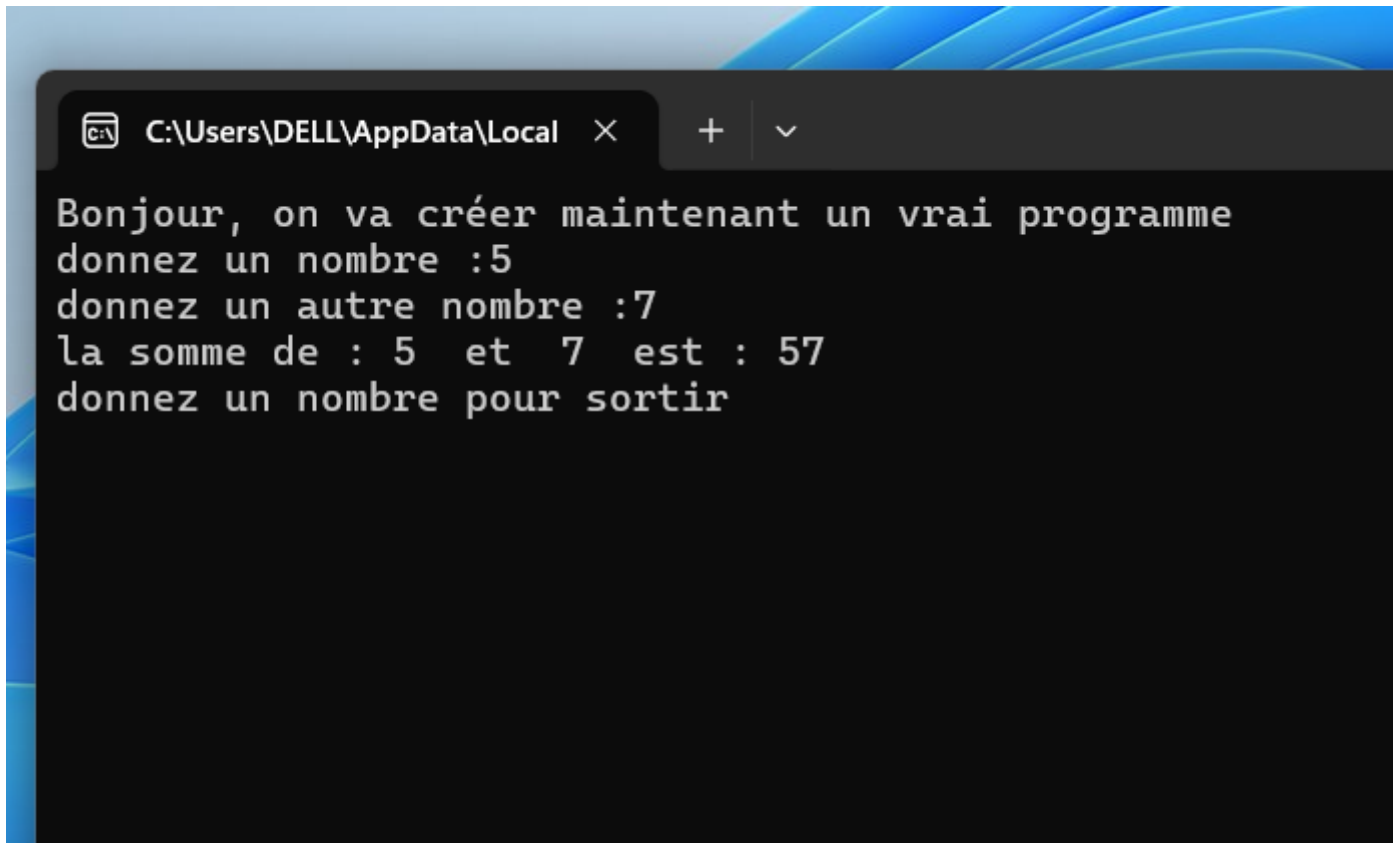
```
Bonjour, on va créer maintenant un vrai programme  
donnez un nombre :5  
donnez un autre nombre :7
```

on a oublié de bloquer l'écran pour voir le résultat !

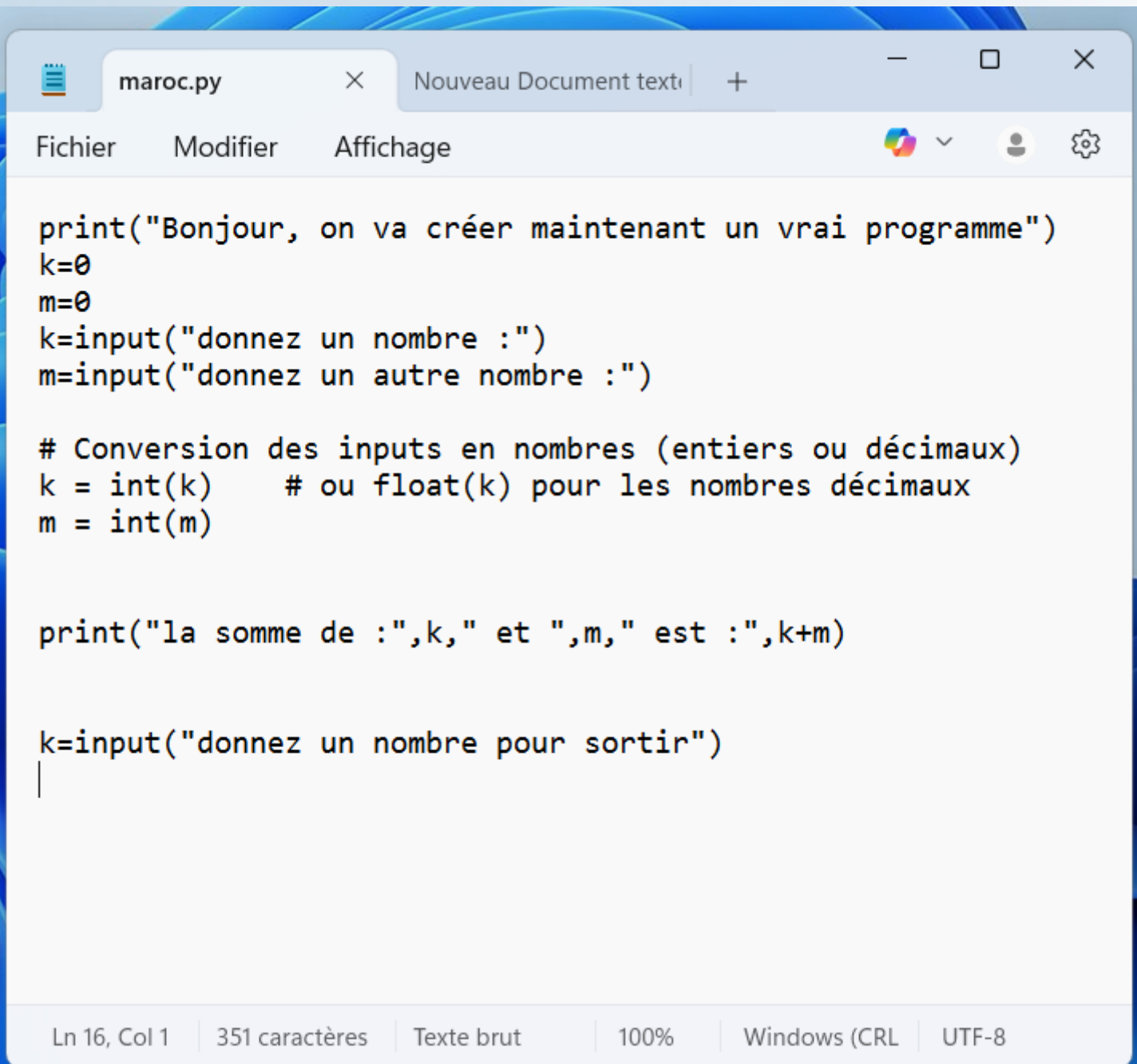


```
maroc.py × +  
Fichier Modifier Affichage  
print("Bonjour, on va créer maintenant un vrai programme")  
k=0  
m=0  
k=input("donnez un nombre :")  
m=input("donnez un autre nombre :")  
  
print("la somme de :",k," et ",m," est :",k+m)  
k=input("donnez un nombre pour sortir")  
  
Ln 11, Col 1 | 223 caractères | Texte brut | 100% | Windows (CRL | UTF-8
```

# Résultat incorrect !!!!



```
C:\Users\DELL\AppData\Local  ×  +  v  
Bonjour, on va créer maintenant un vrai programme  
donnez un nombre :5  
donnez un autre nombre :7  
la somme de : 5 et 7 est : 57  
donnez un nombre pour sortir
```



The image shows a web browser window with a single tab titled 'maroc.py'. The browser's address bar shows 'Nouveau Document text'. The page has a menu bar with 'Fichier', 'Modifier', and 'Affichage'. The main content area contains a Python script. The script starts with a print statement, initializes variables k and m to 0, prompts the user for two numbers, converts them to integers, calculates their sum, and prints the result. It then prompts the user for a number to exit. The status bar at the bottom indicates the cursor is at line 16, column 1, with 351 characters, in 'Texte brut' mode, at 100% zoom, using Windows (CRL) line endings and UTF-8 encoding.

```
print("Bonjour, on va créer maintenant un vrai programme")
k=0
m=0
k=input("donnez un nombre :")
m=input("donnez un autre nombre :")

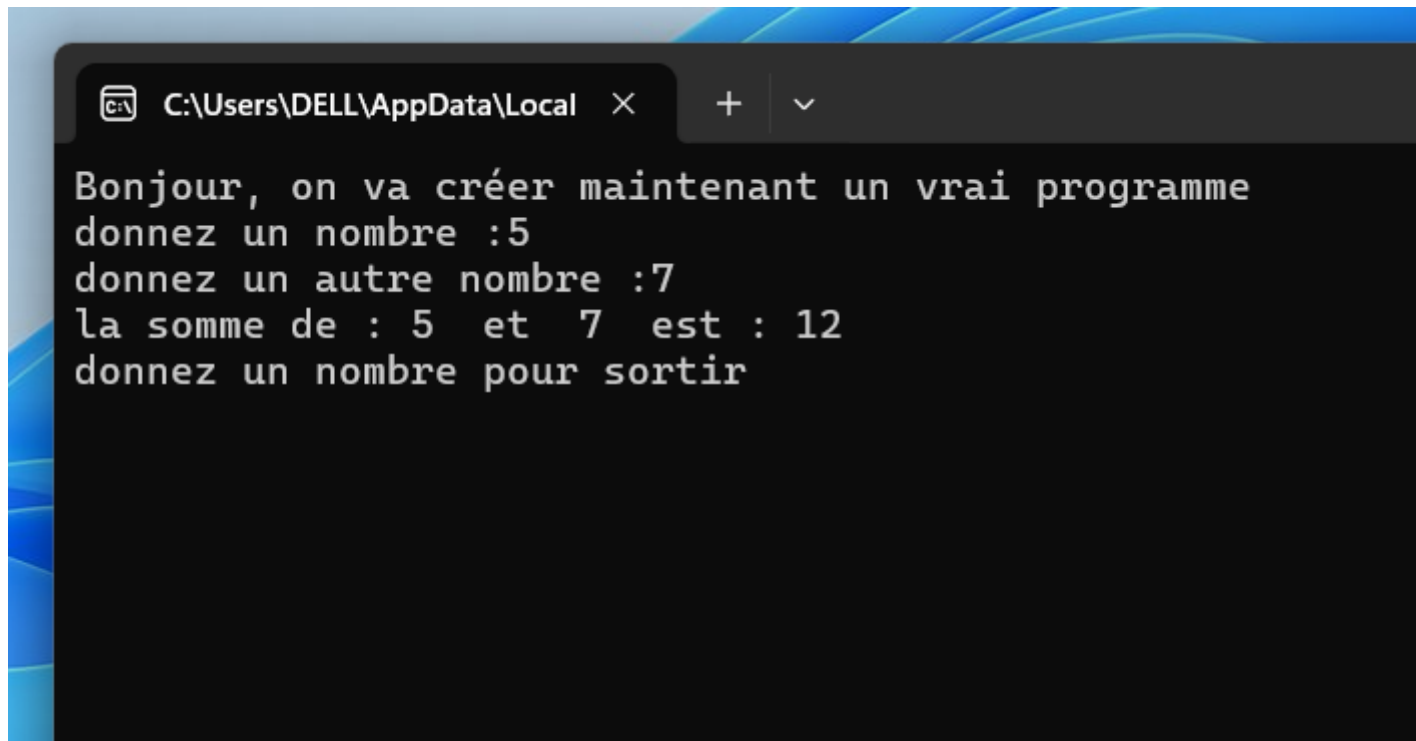
# Conversion des inputs en nombres (entiers ou décimaux)
k = int(k)    # ou float(k) pour les nombres décimaux
m = int(m)

print("la somme de :",k," et ",m," est :",k+m)

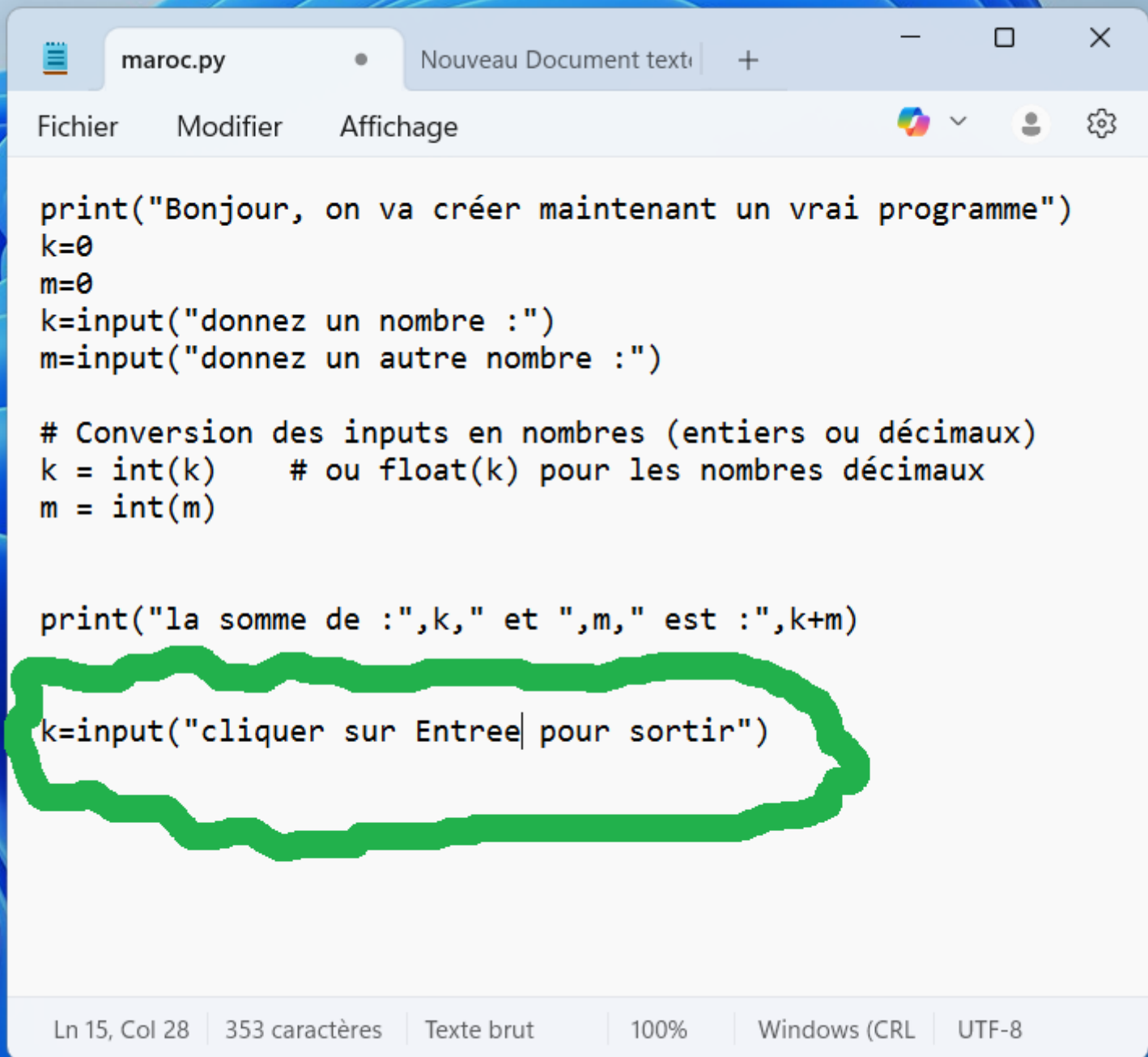
k=input("donnez un nombre pour sortir")
|
```

Ln 16, Col 1 | 351 caractères | Texte brut | 100% | Windows (CRL | UTF-8





```
C:\Users\DELL\AppData\Local × + v  
Bonjour, on va créer maintenant un vrai programme  
donnez un nombre :5  
donnez un autre nombre :7  
la somme de : 5 et 7 est : 12  
donnez un nombre pour sortir
```

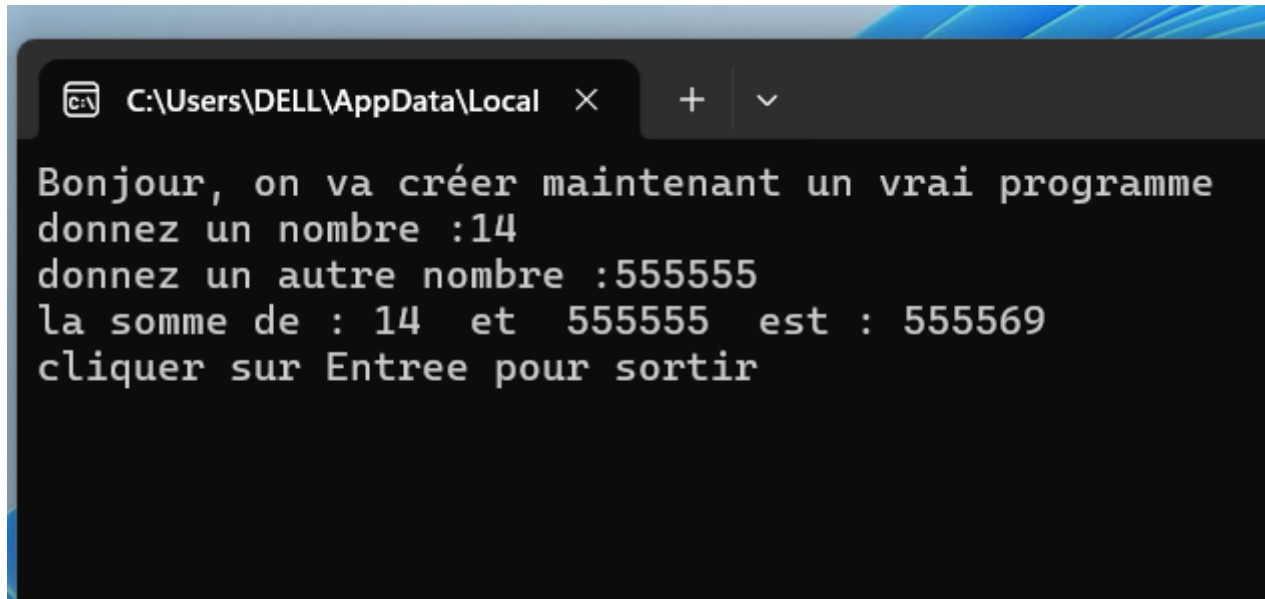


```
print("Bonjour, on va créer maintenant un vrai programme")
k=0
m=0
k=input("donnez un nombre :")
m=input("donnez un autre nombre :")

# Conversion des inputs en nombres (entiers ou décimaux)
k = int(k)    # ou float(k) pour les nombres décimaux
m = int(m)

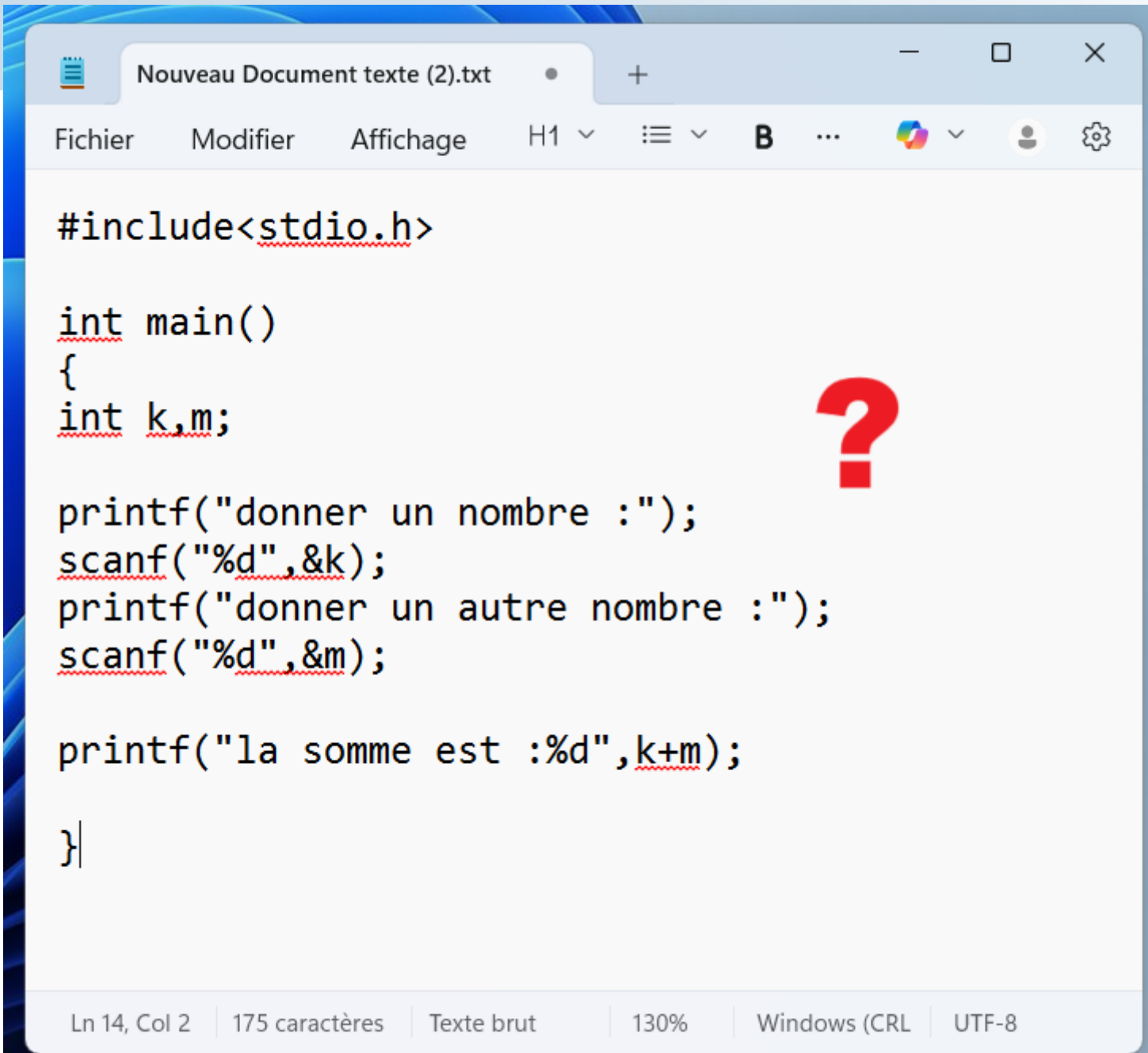
print("la somme de :",k," et ",m," est :",k+m)
k=input("cliquer sur Entree pour sortir")
```

Ln 15, Col 28 | 353 caractères | Texte brut | 100% | Windows (CRL | UTF-8



```
C:\Users\DELL\AppData\Local  ×  +  ∨  
Bonjour, on va créer maintenant un vrai programme  
donnez un nombre :14  
donnez un autre nombre :555555  
la somme de : 14 et 555555 est : 555569  
cliquer sur Entree pour sortir
```

- On peut faire la même chose en utilisant langage C?



The image shows a text editor window titled "Nouveau Document texte (2).txt". The code is written in C and is as follows:

```
#include<stdio.h>

int main()
{
    int k,m;

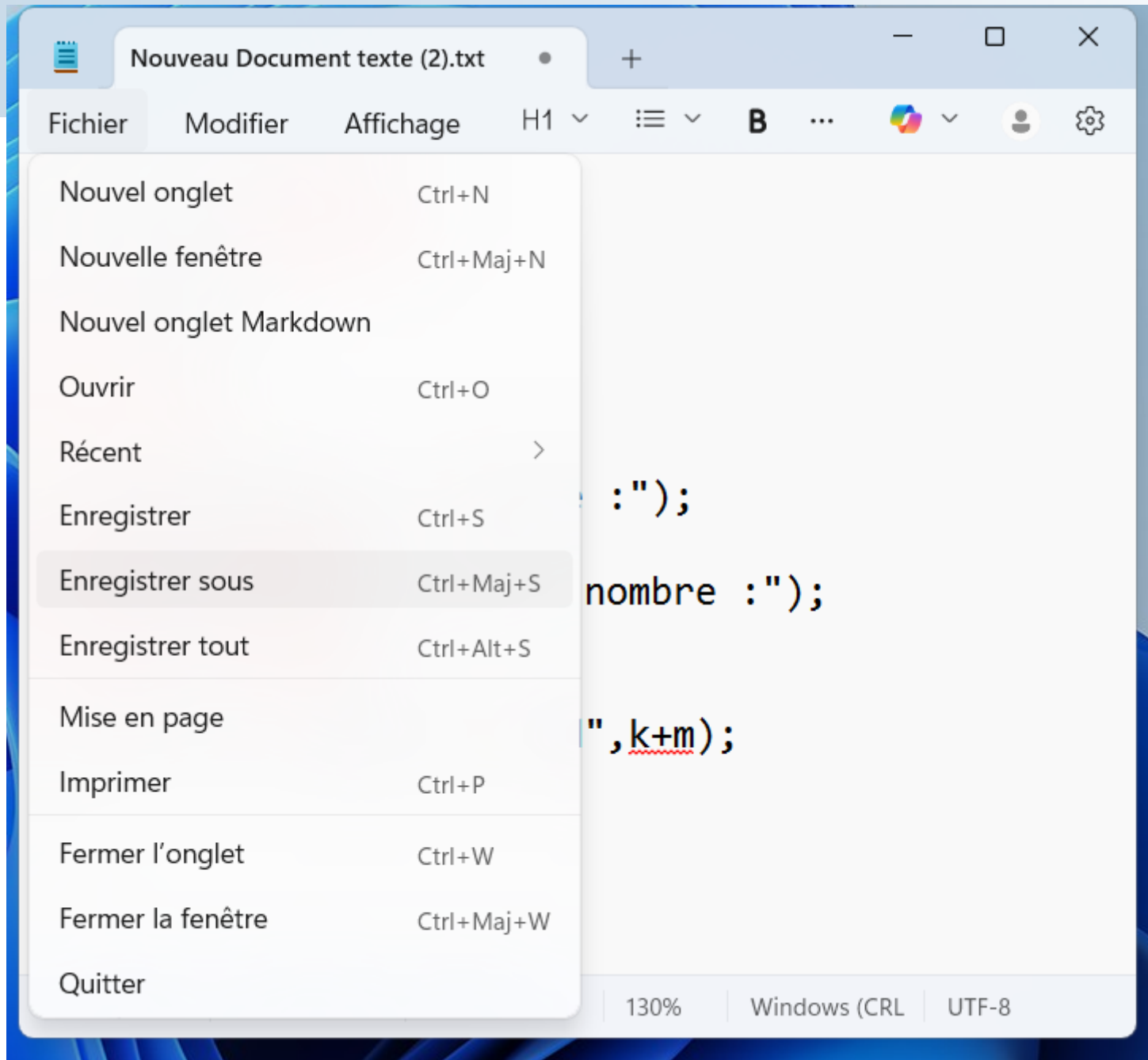
    printf("donner un nombre :");
    scanf("%d",&k);
    printf("donner un autre nombre :");
    scanf("%d",&m);

    printf("la somme est :%d",k+m);

    }
```

A large red question mark is positioned to the right of the code block.

The status bar at the bottom indicates: Ln 14, Col 2 | 175 caractères | Texte brut | 130% | Windows (CRL | UTF-8



# Enregistrer sous



Bureau



Rechercher dans : Bureau

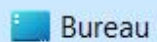


Organiser

Nouveau dossier



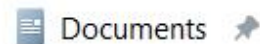
Galerie



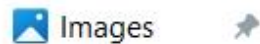
Bureau



Téléchargem



Documents



Images

??

Nom du fichier : maroc.c

Type : Tous les fichiers

Masquer les dossiers

Codage : UTF-8

Enregistrer

Annuler

Ln 14, Col 2

175 caractères

Texte brut

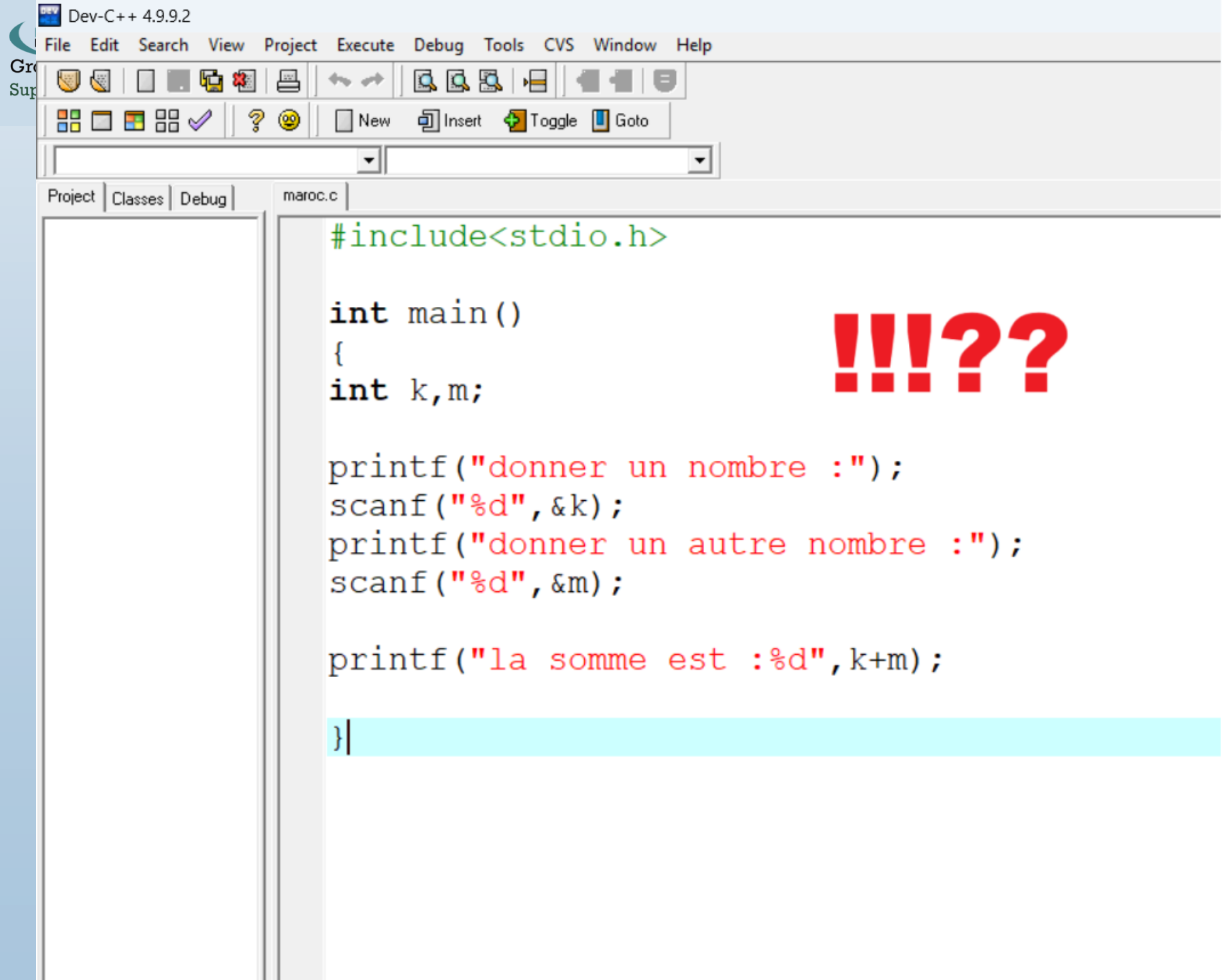
130%

Windows (CRL

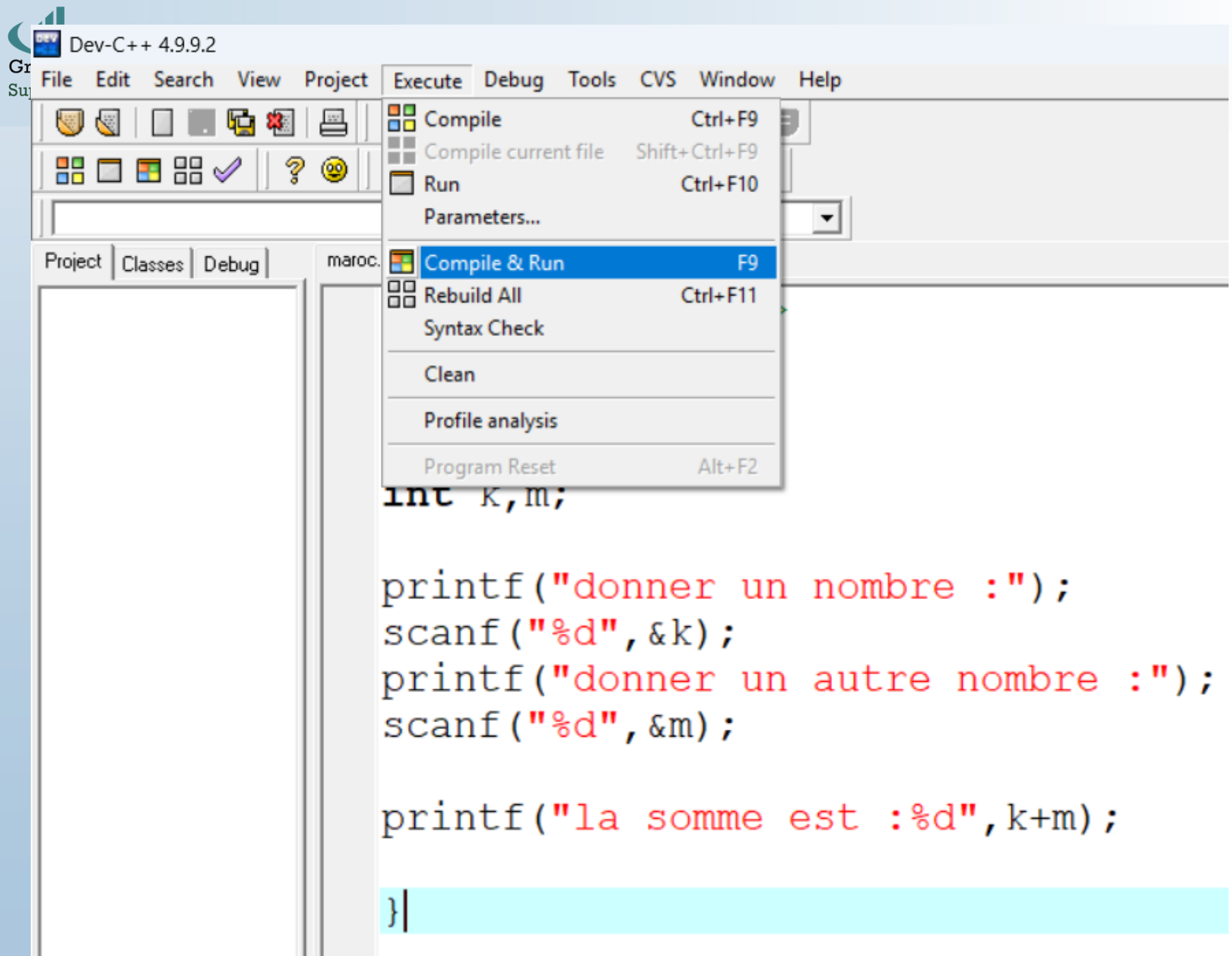
UTF-8







- Donc ?
- Un code en C ne peut pas être interprété directement
- On doit le compiler pour pouvoir l'exécuter



# **Partie 2**

# **Types de variables en Python**

## ■ **Types numériques**

- Type int – Entier
- Type float – Flottant
- Type complex - Complexe

## ■ **Type texte**

- Type str - Chaîne de caractères

## ■ **Types séquentiels**

- Type list – Liste
- Type tuple - Tuple

## ■ **Types de mapping**

- Type dict - Dictionnaire

## ■ **Types d'ensembles**

- Type set - Ensemble

## ■ **Type booléen**

- Type bool - Booléen

## ■ **Types binaires**

- Type bytes – Bytes
- Type bytearray - Tableau de bytes

## Types numériques

Les types suivants représentent des données numériques:

- Type int – Entier
- Type float – Flottant
- Type complex - Complexe

## Type int - Entier

Nombre entier sans virgule

age = 25

quantite = -10

resultat = age + quantite

## Type float - Flottant

Nombre à virgule flottante

prix = 19.99

temperature = -5.5

moyenne = (prix + temperature) / 2



## Type complex - Complexe

Nombre complexe avec partie réelle et imaginaire

$$z_1 = 2 + 3j$$

$$z_2 = 4 - 1j$$

$$\text{somme} = z_1 + z_2$$

Les créateurs de Python ont bien pensé aux nombres complexes mathématiques.

En Python, le type complex représente exactement l'ensemble  $\mathbb{C}$  des nombres complexes des mathématiques.

La seule différence est la notation :

**Mathématiques** :  $a + bi$  (avec  $i$  comme unité imaginaire)

**Python** :  $a + bj$  (avec  $j$  comme unité imaginaire)

Ils ont choisi 'j' car c'est la notation utilisée en ingénierie électrique et en programmation.

Mais les opérations sont les mêmes :

Addition, soustraction, multiplication, division

Pour l'exemple precedent on avait un nombre 'somme'  
donc

Partie réelle (somme.real)

Partie imaginaire (somme.imag)

C'est bien le même concept mathématique !

Le conjugué d'un nombre complexe  $z=a+ib$  est le nombre  $a-ib$ , obtenu en changeant le signe de la partie imaginaire.

Géométriquement, le conjugué est le symétrique du nombre complexe par rapport à l'axe des réels dans le plan complexe

Conjugué (somme.conjugate())

## Type texte

Le type suivant représente du texte

➤ Type str - Chaîne de caractères

## **Type str - Chaîne de caractères**

Séquence de caractères entre guillemets

nom = "Ali"

message = 'Bonjour'

salutation = message + " " + nom

## Types séquentiels

Les types suivants représentent des collections ordonnées

- Type list – Liste
- Type tuple - Tuple

## Type list - Liste

Collection ordonnée et modifiable

```
fruits = ["pomme", "banane", "orange"]
```

```
fruits.append("kiwi")
```

```
premier = fruits[0]
```

## Type tuple - Tuple

Collection ordonnée et non modifiable

```
coordonnees = (10, 20)
```

```
couleur_rgb = (255, 0, 0)
```

```
x = coordonnees[0]
```



## Types de mapping

Le type suivant représente des associations clé-valeur

➤ Type dict - Dictionnaire

## Type dict - Dictionnaire

Collection de paires clé-valeur

```
etudiant = {"nom": "Pierre", "age": 20}
```

```
etudiant["note"] = 15
```

```
nom_etudiant = etudiant["nom"]
```

## Types d'ensembles

Les types suivants représentent des collections d'éléments uniques

➤ Type set - Ensemble

## Type set - Ensemble

Collection non ordonnée d'éléments uniques

```
nombres = {1, 2, 3, 3, 2}
```

```
nombres.add(4)
```

```
nombres.remove(1)
```

## Type booléen

Le type suivant représente des valeurs logiques

➤ Type bool - Booléen

## Type bool - Booléen

Valeur logique True ou False

```
est_actif = True
```

```
est_vide = False
```

```
resultat = est_actif and est_vide
```

## Types binaires

Les types suivants représentent des données binaires

- Type bytes – Bytes
- Type bytearray - Tableau de bytes

## Type bytes - Bytes

Séquence d'octets immuable

```
data1 = b"SUPMTI"
```

```
data2 = bytes([83, 85, 80, 77, 84, 73])
```



## Explication :

- $S \rightarrow \text{code ASCII } 83$
- $U \rightarrow \text{code ASCII } 85$
- $P \rightarrow \text{code ASCII } 80$
- $M \rightarrow \text{code ASCII } 77$
- $T \rightarrow \text{code ASCII } 84$
- $I \rightarrow \text{code ASCII } 73$

Donc  $\text{data1} = \text{data2}$

## Type bytearray - Tableau de bytes

Séquence d'octets modifiable

```
buffer = bytearray(5)
```

```
buffer[0] = 65
```

```
buffer[1] = 66
```

## **Partie 3 : Structures conditionnelles et Structures itératives**

## **Structures conditionnelles**

Les instructions qui permettent d'exécuter du code selon des conditions

## Structure if - Si simple

Exécute un bloc si la condition est vraie

```
age = 18
```

```
if age >= 18:
```

```
    print("Majeur")
```

## Structure if-else - Si sinon

Exécute un bloc si vrai, un autre si faux

```
note = 12
```

```
if note >= 10:
```

```
    print("Validé")
```

```
else:
```

```
    print("Non validé")
```

## Structure if-elif-else - Si sinon si

Teste plusieurs conditions successivement

```
note = 15
```

```
if note >= 16:
```

```
    print("Très bien")
```

```
elif note >= 14:
```

```
    print("Bien")
```

```
elif note >= 12:
```

```
    print("Assez bien")
```

```
else:
```

```
    print("Passable")
```

## Structures itératives

Les instructions qui répètent l'exécution d'un bloc de code



## **Boucle for - Pour**

Répète pour chaque élément d'une séquence

```
fruits = ["pomme", "banane", "orange"]
```

```
for fruit in fruits:
```

```
    print(fruit)
```

## **Boucle while - Tant que**

Répète tant qu'une condition est vraie

```
compteur = 0
```

```
while compteur < 5:
```

```
    print(compteur)
```

```
    compteur += 1
```

## Partie 4 : Fonctions

# Fonctions

Blocs de code réutilisables avec des paramètres

## Définition de fonction

Crée une fonction avec def et des paramètres

```
def addition(a, b):
```

```
    resultat = a + b
```

```
    return resultat
```

## Appel de fonction

Utilise une fonction avec des arguments

```
somme = addition(5, 3)
```

```
print(somme) # Affiche 8
```

## **Partie 5 : l'équivalent des tableaux en Python ?**

```
#include <stdio.h>
```

```
int main() {
```

```
    int nombres[5] = {10, 20, 30, 40, 50};
```

```
    printf("%d\n", nombres[0]);
```

```
    nombres[1] = 25;
```

```
    for(int i = 0; i < 5; i++) {
```

```
        printf("%d ", nombres[i]);
```

```
    }
```

```
    return 0; }
```



## L'équivalent en Python (liste) :

```
nombres = [10, 20, 30, 40, 50]
```

```
print(nombres[0])
```

```
nombres[1] = 25
```

```
for nombre in nombres:
```

```
    print(nombre, end=" ")
```

## **Partie 6 : l'équivalent des structures en Python ?**

## Structure en C :

```
#include <stdio.h>
```

```
struct Etudiant {
```

```
    char nom[50];
```

```
    int age;
```

```
    float note;    };
```

```
int main() {
```

```
    struct Etudiant etudiant = {"Mohamed", 20, 15.50};
```

```
    printf("%s\n", etudiant.nom);
```

```
    return 0;    }
```

les dictionnaires Python sont la structure qui se rapproche le plus des struct en C.

## **Python (dictionnaire) :**

```
etudiant = {"nom": " Mohamed ", "age": 20, "note": 15}  
print(etudiant["nom"])
```

## **Partie 7 : Définition simple d'une classe et son utilisation**

**ATTENTION : ce n'est pas un cours, c'est juste un rappel + un exemple de classe, vous n'allez pas comprendre quelques mots clés mentionnés dans le prochain exemple, pas grave, on le verra dans le cours prochainement**

**Voici la définition la plus simple d'une classe en Python :**

**Définition simple d'une classe :**

```
class NomDeLaClasse:

    def __init__(self, parametres):

        # Initialisation des attributs

        self.attribut = valeur


    def methode(self):

        # Code de la méthode

        pass
```

# exemple avec la classe Etudiant

```
class Etudiant:
```

```
    def __init__(self, nom, age, note):    # nom, age et note sont des attributs
```

```
        self.nom = nom
```

```
        self.age = age
```

```
        self.note = note
```

```
    def afficherMesInfos(self):
```

```
        print(f"Nom: {self.nom}, Age: {self.age}, Note: {self.note}")
```

**# Création de 2 étudiants (instanciation de 2 étudiants)**

```
etud1 = Etudiant("Ali", 20, 15.5)
```

```
etud2 = Etudiant("Rym", 22, 14.0)
```

**# Appel de leurs méthodes**

```
etud1.afficherMesInfos()
```

```
etud2.afficherMesInfos()
```