

LE POLYTECHNIQUE DE L’UNIVERSITE FRANÇOIS RABELAIS DE TOURS

Spécialité Informatique

64 avenue Jean Portalis,

37200 TOURS, FRANCE

Tél +33 (0)2 47 36 14 14

[www.polytech.univ-tours.fr](http://www.polytech.univ-tours.fr/)

Rapport Python

**FENG Jiaming**

**21707371**

**XU Tianjiao**

**21707404**

**Table des matières**

[Avant-propos 3](#_Toc28010)

[TP1-Fichiers 3](#_Toc21720)

[TP2-Tkinter 8](#_Toc31794)

[1. Introduction 8](#_Toc19364)

[Entry 8](#_Toc12668)

[Button 9](#_Toc21856)

[Eval 9](#_Toc13212)

[2. Travail du TP 10](#_Toc7325)

[TP3-Exceptions et chiffrement 13](#_Toc11432)

[1. Exceptions 13](#_Toc17637)

[2. Travail du TP 14](#_Toc31135)

[TP4 Matplotlib 19](#_Toc9678)

# Avant-propos

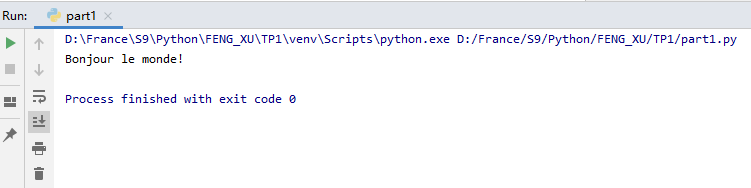
Les codes du TP sont fournis dans le fichier Code\_Python.zip .

Au cours de TP, nous utilisons PyCharm comme IDE.

La version de python est 3.7.1

# TP1-Fichiers

**1. Afficher « Bonjour le monde ! »**



**2. En mode console, proposez un petit menu :**

**1. Choisir un nom de fichier,**

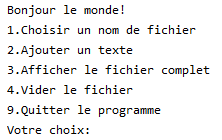
**2. Ajouter un texte (que vous demanderez à l’utilisateur),**

**3. Afficher le fichier complet,**

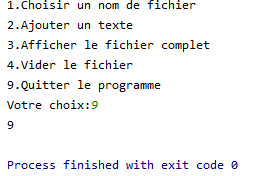
**4. Vider le fichier,**

**9. Quitter le programme.**

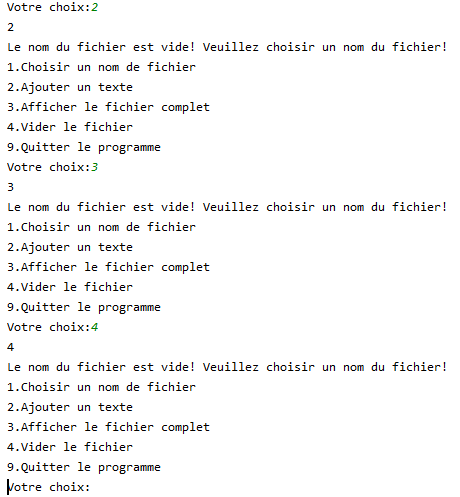
L'interface du menu est la suivante：



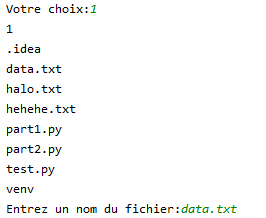
Entrez 9 pour quitter le programme：



Ce programme doit d'abord sélectionner un nom de fichier. Si aucun autre fichier n'est sélectionné, l'utilisateur sera invité à entrer le nom du fichier. Comme indiqué ci-dessous:



L’interface d’exécution 1 est la suivante:

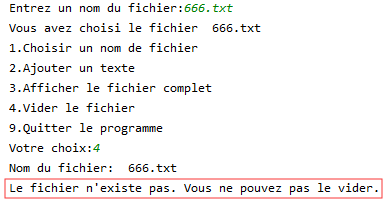
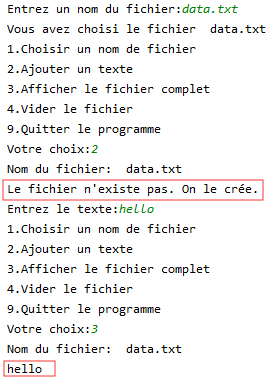


Nous utilisons les fonctions "**listdir**" et "**getcwd**" de os pour lister les fichiers du répertoire actuel que les utilisateurs peuvent choisir.

**for** f **in** os.listdir(os.getcwd()):  
 print(f)

L'utilisateur peut sélectionner un fichier existant ou entrer un nouveau nom de fichier.

Après cela, nous exécutons l'option 2 ou 3. Si le fichier n'existe pas un nouveau fichier est créé



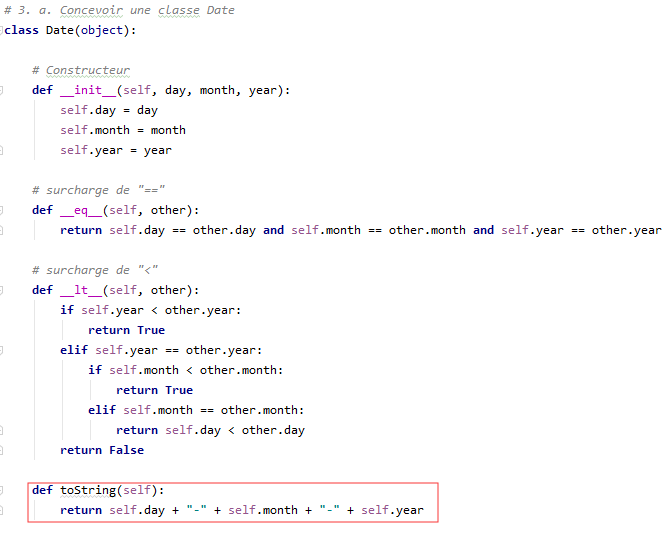
L'option d'exécution 4 est également la même, Si le fichier n'existe pas, il ne fonctionnera pas.

**a. Concevoir une classe Date (avec redéfinition (surcharge) de == ( \_\_eq\_\_ ) et de < ( \_\_lt\_\_ )) ;**

**b. Concevoir une clase Etudiant (avec une méthode adresselec qui fabrique l'adresse électronique prenom.nom@etu.univ-tours.fr et une méthode âge) ;**

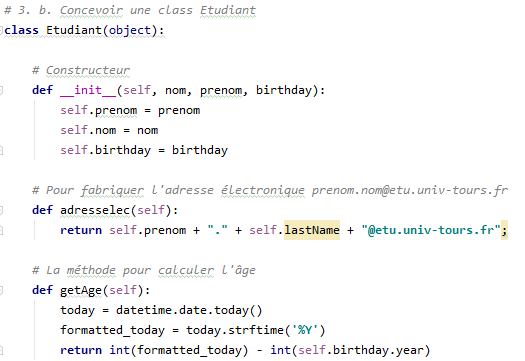
**c. Lire le fichier fichetu.csv et constituer une liste d'objets Etudiant**

3.a. La classe Date est construite comme suit.Nous avons également ajouté une méthode tostring pour renvoyer les attributs date, mois et jour de la classe Date sous forme de chaîne.

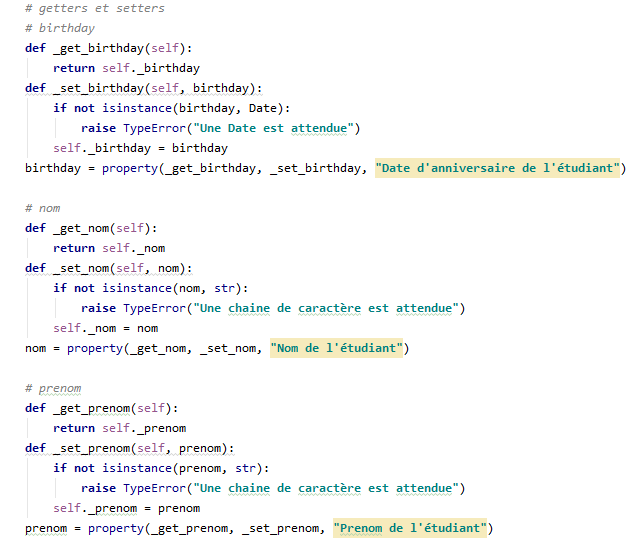


3.b. La classe Etudiant avec une méthode adresselec et une méthode âgeest construite comme suit. Nous avons constaté que lorsque nous utilisons la méthode print pour afficher l'objet Item, nous produisons en fait la valeur de retour de la méthode ***\_\_repr \_\_ ()*** de l'objet Item.

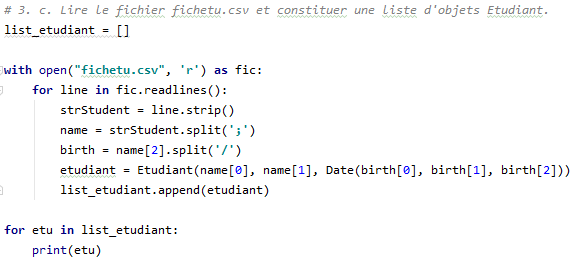
Par conséquent, nous essayons de réécrire la méthode ***\_\_repr \_\_ ()*** pour afficher les informations spécifiques de l'objet etudiant.



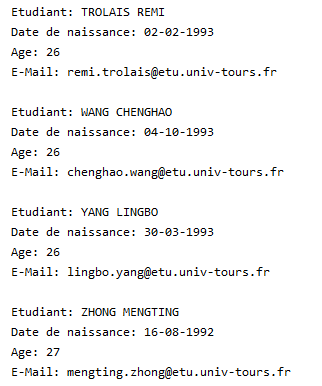
Nous avons également défini "**getter**" et "**setter**" pour la classe "Etudiant". Nous utilisons la fonction "***property***" pour envelopper la méthode en tant qu'attribut afin de pouvoir y accéder et l'appeler en tant que propriété.



3.c. Programme principal pour la lecture de fichiers:



La sortie est montrée ci-dessous:



# TP2-Tkinter

## Introduction

Tkinter (de l’anglais Tool kit interface) est la bibliothèque graphique libre d’origine pour le langage Python, permettant la création d’interfaces graphiques. Tkinter est intégré au paquet d’installation de python pour gérer rapidement des interfaces graphiques simples.

Le premier test Tkinter:



Tkinter offre une variété de composant d’interface graphique tels que ***Boutons , Label , Frame , Canvas , Text , Menu , Message , Scrollbar*** ,etc.

Il possède des attributs standards comme: ***Dimension , Color , Font , Cusor , Anchor*** , etc.

### Entry

Les widgets d'entrée sont les widgets de base de Tkinter utilisés pour obtenir une entrée, c'est-à-dire des chaînes de texte, de l'utilisateur d'une application.

La syntaxe d'un widget d'entrée ressemble à ceci:

w = entry(master, option, ...)

"master" représente la fenêtre parente, où le widget de saisie doit être placé. Comme pour d'autres widgets, il est possible d'influencer davantage le rendu du widget en utilisant des options. La liste d'options séparées par des virgules peut être vide.

Nous utilisons principalement les options suivantes:

* **Justify:** Lorsque vous affichez plusieurs lignes de texte, définissez l’alignement entre différentes lignes, notamment LEFT, RIGHT, CENTER
* **State:** La valeur par défaut est state = NORMAL, l’état de la boîte, divisé en lecture seule et en écriture, la valeur est: normal / désactivé
* **Textvariable:** La valeur de la zone de texte est un objet StringVar ()
* **Xscrollcommand:** Définissez la barre de défilement horizontale, qui est généralement utilisée lorsque la largeur de la zone de texte saisie par l'utilisateur est supérieure à la largeur de la zone de texte.

**Les méthodes:**

* **get():** Obtenir la valeur de la boîte de fichier
* **delete ( first, last=None ) :** Supprimer la valeur de position directe dans la zone de texte

### Button

Les composants Button permettent d’ajouter des boutons à une application Python, des boutons peuvent être placés avec du texte ou des images, des boutons permettent d’écouter le comportement de l’utilisateur, peuvent être associés à une fonction Python et sont automatiquement appelés lorsque le bouton est enfoncé.

La syntaxe est la suivante:

w = Button ( master, option=value, ... )

master: le conteneur parent du bouton.

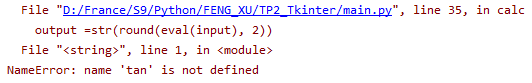
**Les options principales**:

* **Activebackground**: La couleur de fond du bouton lorsque la souris est placée
* **Activeforeground**: La couleur de premier plan du bouton lorsque la souris est placée
* **Bg**: La couleur de fond du bouton
* **Command**: La fonction associée au bouton, qui est exécutée lorsque l'utilisateur clique sur le bouton
* **Fg**: La couleur de premier plan du bouton (la couleur du texte du bouton)

### Eval

La fonction eval () est utilisée pour exécuter une expression de chaîne et renvoyer la valeur de l'expression.

**Note**: Lorsque nous utilisons la fonction eval pour exécuter les fonctions tan, cos, sin, nous constatons que python signalera l'erreur suivante.

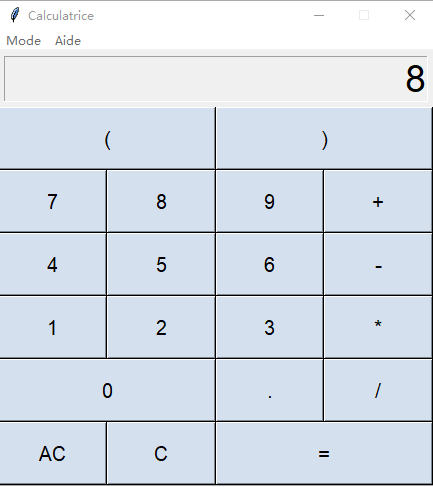


Après des recherches, on’a découvert que il faut importer la bibliothèque de mathématiques.

## Travail du TP

1. **Faire une calculatrice (10 chiffres, 4 opérations, =, C, AC) qui récupère dans une chaîe les touches cliquées par l’utilisateur. C efface la dernière touche cliquée et AC efface toutes les touches bufferisées.**

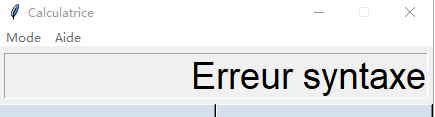
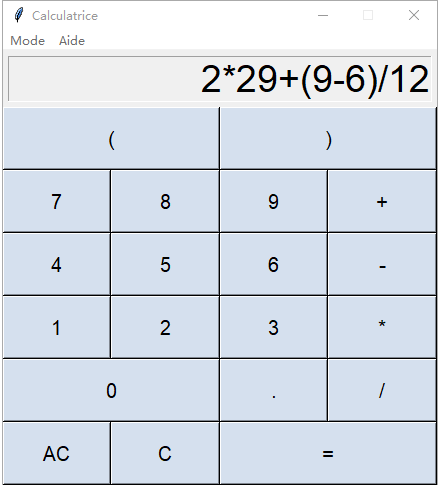
L'interface de calculatrice que nous avons conçue est la suivante.



Nous utilisons principalement les composants **Entry** et **Button** pour construire l’interface de la calculatrice. Pour normaliser la mise en page, nous utilisons la fonction de **grid**().

On utilise les paramètres **row**, **column,sticky,columnspan** pour déterminer l'emplacement du composant.

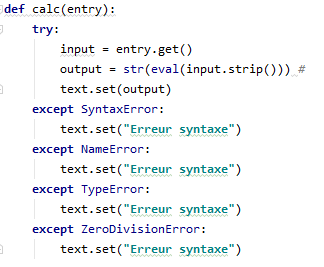
**2. Résoudre le calcul et afficher le résultat (vous pouvez utiliser la fonction eval() )**

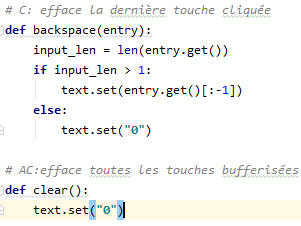


Nous utilisons la fonction "**get\_input**" pour obtenir la saisie de la calculatrice.

**def** get\_input(entry, argu)

Après avoir cliqué sur le bouton "=", il appelle la fonction "calc" qui utilise la fonction eval pour exécuter une expression de chaîne et renvoyer la valeur de l'expression.

Si l'expression signale une erreur, "Erreur syntaxe" apparaîtra. AC et C sont implémentés comme suit:



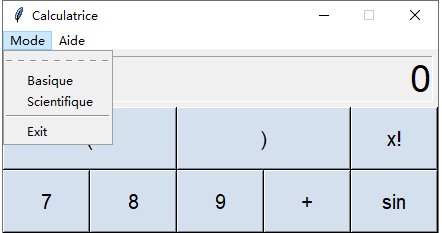
On peut observer que nous utilisons la méthode **text.set** pour contrôler la sortie.

En effet, nous définissons le composant de **Entry** avec l’attribut **readonly**. Nous pouvons uniquement modifier la propriété textvariable pour modifier le contenu de la zone de texte.

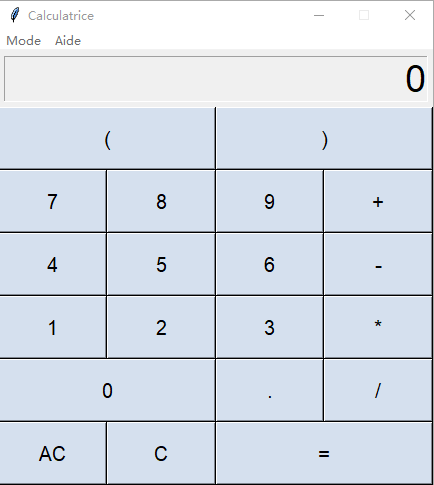
entry = Entry(root, textvariable=text, justify=**"right"**, font=(**"Arial"**, 28), state=**"readonly"**)

**3. Ajouter un menu avec des commandes de votre choix (aide, mode basique/scientifique…)**

Nous avons créé la barre de menu selon les besoins en utilisant le menu des composants de tkinter.

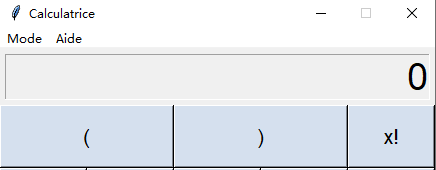


L'interface entre les deux modes est la suivante:

La mode scientifique ajoute d'une nouveau colonne de bouton. Des fonctions telles que «**sin**» ,«**cos**» et «**tan**» seront directement émises dans la zone de saisie et calculées par la fonction eval. Des fonctions «**CarréRacine**», «**carrée**» et «**factorielle**» sont implémentées en appelant la forme de la bibliothèque «math».

**4.Ajouter les parenthèses**

Les parenthèses peuvent également être identifiées par la fonction eval.

**5. Continuer d’améliorer votre calculatrice (décimale, puissance, trigo…)**

Toutes les nouvelles fonctionnalités sont ajoutées au modèle scientifique et fonctionnent correctement.

Notre calculatrice peut également émettre des jugements simples sur l'entrée pour corriger l'expression.

# TP3-Exceptions et chiffrement

## Exceptions

Python lève donc des exceptions quand il trouve une erreur, soit dans le code (une erreur de syntaxe, par exemple), soit dans l'opération que vous lui demandez de faire.

Nous allons en effet mettre les instructions que nous souhaitons tester dans un premier bloc et les instructions à exécuter en cas d'erreur dans un autre bloc.

**try:**

**# Bloc à essayer**

**except:**

**# Bloc qui sera exécuté en cas d'erreur**

On peut capturer l'exception et afficher son message grâce au mot-cléasque vous avez déjà vu dans un autre contexte

**try:**

**# Bloc de test**

**except type\_de\_l\_exception as exception\_retournee:**

**print("Voici l'erreur :", exception\_retournee)**

Les mots-clés **else** et **finally**

Ce sont deux mots-clés qui vont nous permettre de construire un bloctryplus complet.

Le mot-clé **else**

Vous avez déjà vu ce mot-clé et j'espère que vous vous en rappelez. Dans un bloc try,else va permettre d'exécuter une action si aucune erreur ne survient dans le bloc.

Le mot-clé **finally**

Finally permet d'exécuter du code après un bloctry, quel que soit le résultat de l'exécution dudit bloc. La syntaxe est des plus simples :

**try:**

**# Test d'instruction(s)**

**except type\_de\_l\_exception:**

**# Traitement en cas d'erreur**

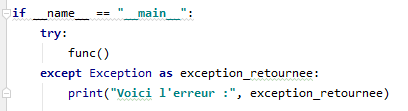
**finally:**

**# Instruction(s) exécutée(s) qu'il y ait eu des erreurs ou non**

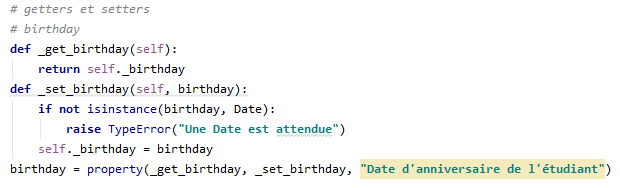
## Travail du TP

1. **Reprenez vos programmes du TP1 (menu) et TP2 (calculatrice) et ajouter la gestion des exceptions (try, except, else, finally).**

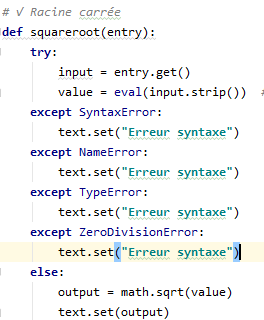
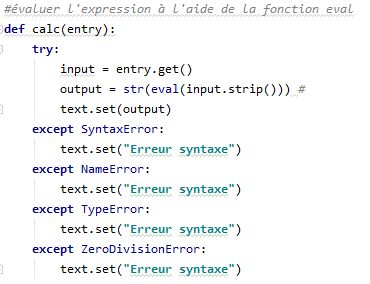
Pour le TP1, Nous avons ajouté try bloc au programme de menu.



Pour les méthodes getters et setters de la classe etudiant, on va lever une exception pour vérifier le type de valeur.



Pour le calculatrice, on’a ajouté le bloc try...except pour capturer de différents types d'exceptions. Comme **SyntaxError**, **NameError**, **TypeError** ou **ZeroDivisionError**



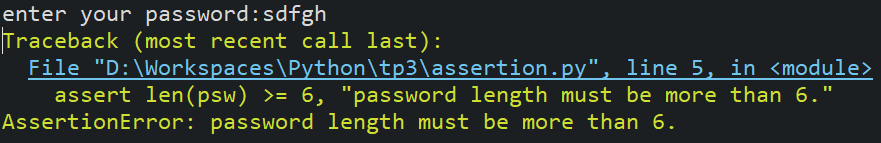
Le mot-clé **else** est également utilisé dans la méthode squareroot.

1. **Lever une exception avec assert et raise.**

Nous proposons une hypothèse pour juger de la longueur du mot de passe.

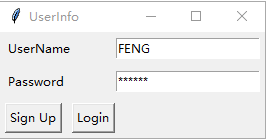
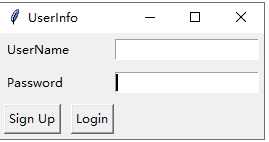
psw = input("enter your password:")

assert len(psw) >= 6, "password length must be more than 6."

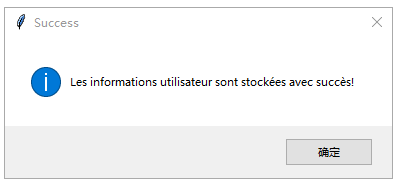


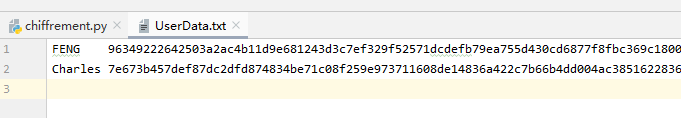
1. **Demander pour enregistrement un login (input) et un mot de passe à l’utilisateur avec getpass() en mode console ou avec une fenêtre tkinter affichant des \* à la place des caractères saisis. Stocker le couple login et hash mot de passe dans un fichier texte. Demander pour vérification un login et un mot de passe, vérifier la présence dans le fichier. Utiliser le « salage » pour renforcer le système avec une chaîne comprenant le login et une partie fixe.**

Nous avons choisi de créer une fenêtre tkinter comme celle ci-dessous.

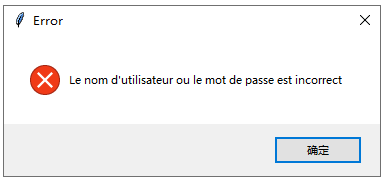
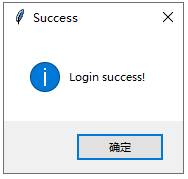


Nous pouvons observer le champ de saisie du mot de passe pour afficher \* à la place des caractères saisis. Après avoir cliqué sur "Sign Up", une boîte de dialogue apparaîtra et l'enregistrement se fera avec succès. Le couple login et hash mot de passe est stocké dans un fichier texte.

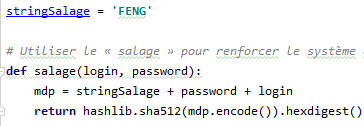




Quand on va demander pour vérification un login et un mot de passe, on vérifier la présence dans le fichier. Les résultats du succès et de l'échec de la connexion sont les suivants:

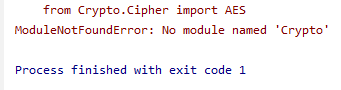


On n’a aussi utilisé un « salage » pour renforcer le système avec une chaîne comprenant le login et une partie fixe.



1. **Demander à l’utilisateur le nom d’un fichier existant (par ex. un fichier texte que vous avez créé) que vous allez chiffrer dans un nouveau fichier en utilisant le mot de passe précédent avec l’algorithme AES 256. Vous pouvez utiliser la bibliothèque PyCryptodome http://pycryptodome.readthedocs.io/en/latest/ ou une autre si elle pose problème. Proposer aussi le déchiffrement du fichier.**

Lorsque nous importons le package crypto, nous obtenons l’erreur suivante.



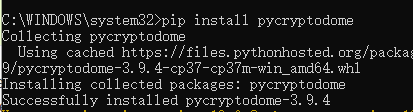
La cause fondamentale en est que, dans le passé, vous avez probablement installé un paquet appelé crypto, non apparenté mais portant le même nom, qui opère sous le crypto d'espace de noms.

Nous résolvons le problème de la manière suivante:

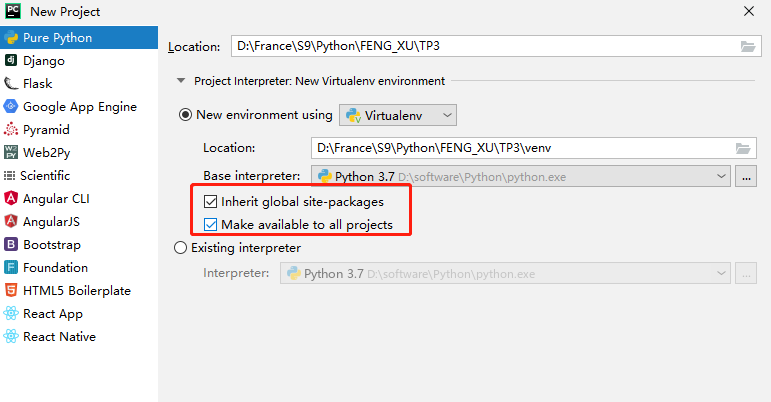
pip uninstall crypto

pip uninstall pycryptodome

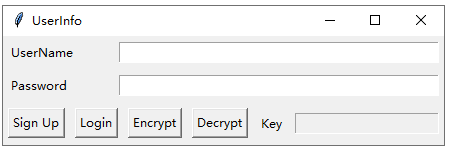
pip install pycryptodome



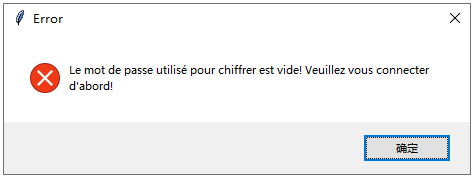
Il faut également vérifier les deux options suivantes lors de la création d'un nouveau projet.



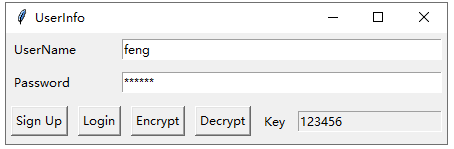
La nouvelle interface est indiquée ci-dessous. Nous avons ajouté deux nouveaux boutons "Cryptage" et "Décryptage" et une zone de texte indiquant la clé de cryptage.



Pour chiffrer un fichier, vous devez d'abord vous connecter pour obtenir le mot de passe pour le chiffrement. Sinon, une erreur se produira.



Une fois connecté, le mot de passe de l'utilisateur s'affiche dans la zone de texte située dans le coin inférieur droit. Après cela, nous chiffrerons le fichier avec le mot de passe de connexion de l'utilisateur.



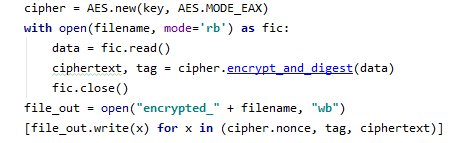
L'algorithme que nous utilisons pour chiffrer est AES256. Nous avons donc besoin d'une clé 32 bits(16 bits pour AES128).

Pour ce faire, nous avons choisi d'utiliser la fonction pad de la crypto-bibliothèque pour compléter la clé de cryptage sur 32 bits pour utiliser l'algorithme AES256

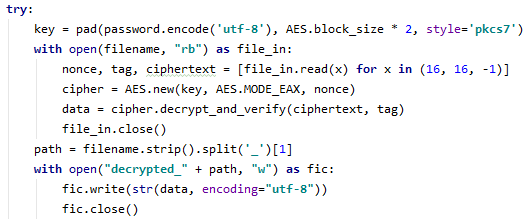
**from** Crypto.Util.Padding **import** pad

key = pad(password.encode(**'utf-8'**), AES.block\_size \* 2, style=**'pkcs7'**)

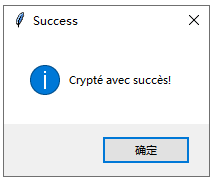
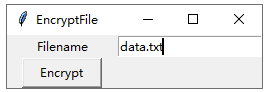
Le code de base pour chiffrer un fichier est le suivant:



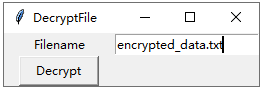
Le code pour décrypter le fichier est le suivant:



Après avoir cliqué sur “Encrypt”, une nouvelle fenêtre s’ouvrira pour indiquer le nom du fichier à chiffrer.

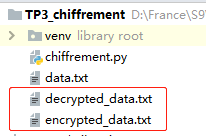


La fenêtre de décryptage est presque la même

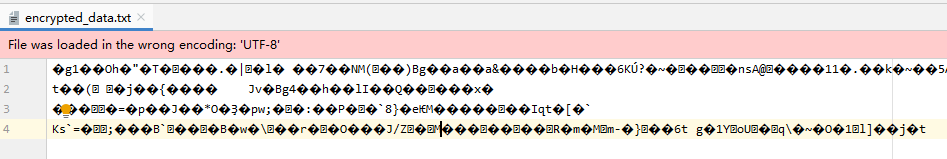


Le contenu du fichier crypté sera stocké dans un fichier portant le préfixe "Encrypted".

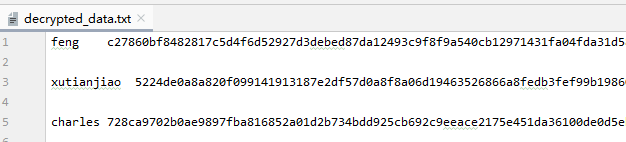
Le préfixe du fichier déchiffré est nommé "Decrypted"



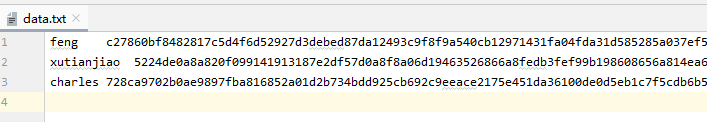
Fichier crypté:



Fichier décrypté:



Fichier source:



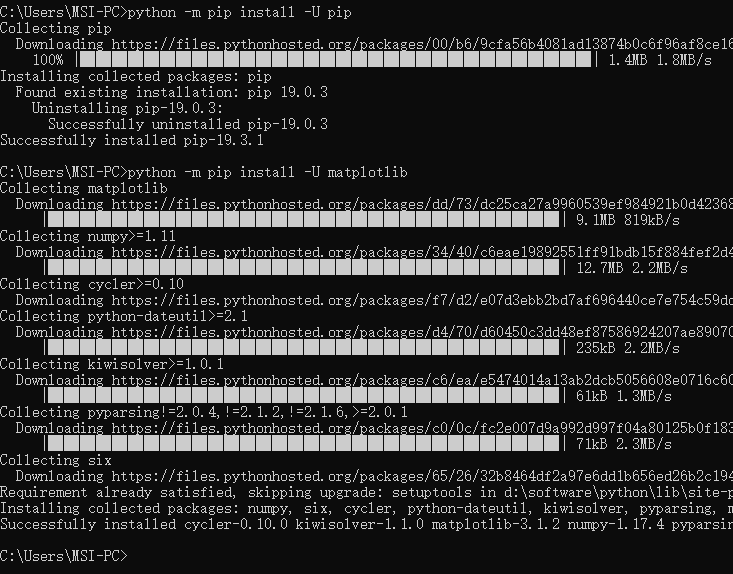
# TP4 Matplotlib

## Matplotlib

Avant de commencer ce TP, nous devons d'abord télécharger la bibliothèque.

python -m pip install -U pip

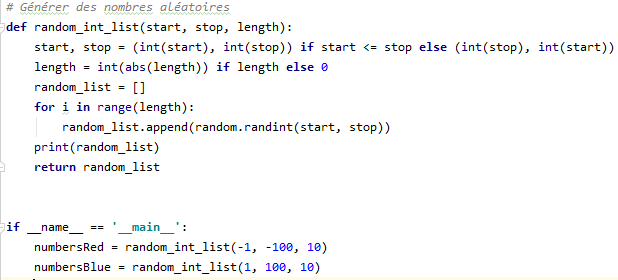
python -m pip install -U matplotlib



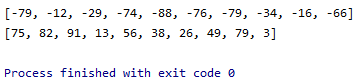
## Travail du TP

1. **Générer des nombres aléatoires**

Nous définissons une fonction random\_init\_list qui qui peut spécifier la plage et le nombre de données pour implémenter la génération de nombres aléatoires.

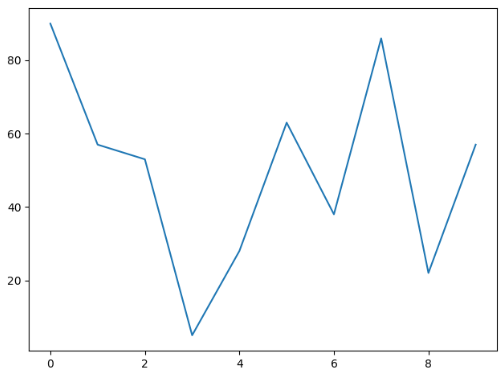


Nous avons généré les deux ensembles de données suivants：



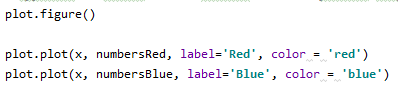
1. **Afficher la courbe de ces données dans une fenêtre matplotlib**

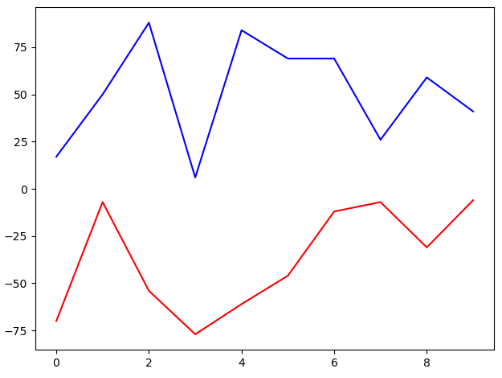
Nous utilisons la méthode **plot** de la bibliothèque matplotlib pour dessiner la courbe.



1. **Afficher plusieurs courbes avec styles et couleurs variés**

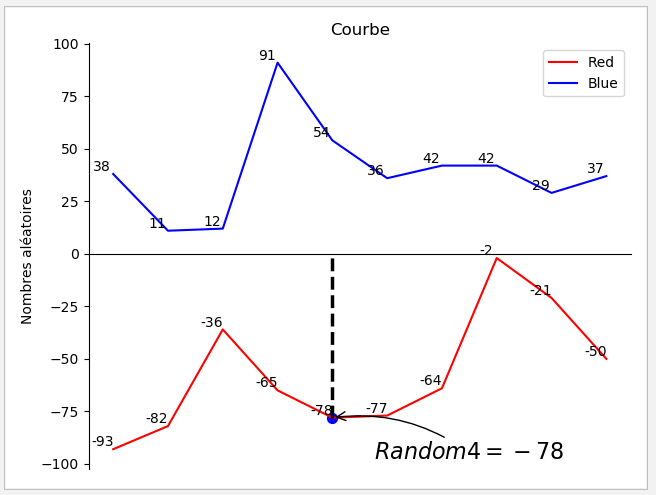
Nous modifions les paramètres de la fonction de tracé pour déterminer les styles et couleurs de la courbe.





1. **Modifier les noms des axes, la légende, ajouter des flèches pour montrer des zones…**

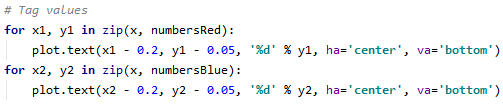
Ce qui suit est un graphique traité, nous avons ajouté les noms des axes, la légende et les flèches pour montrer des zones.

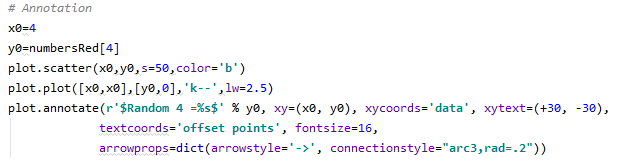


La fonction utilisée est la suivante:





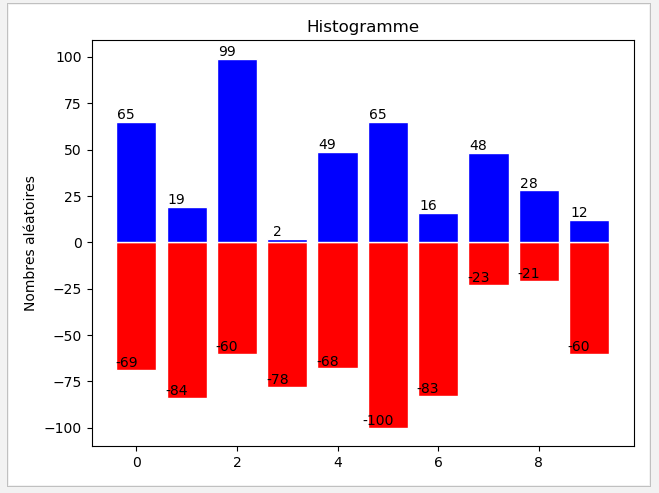




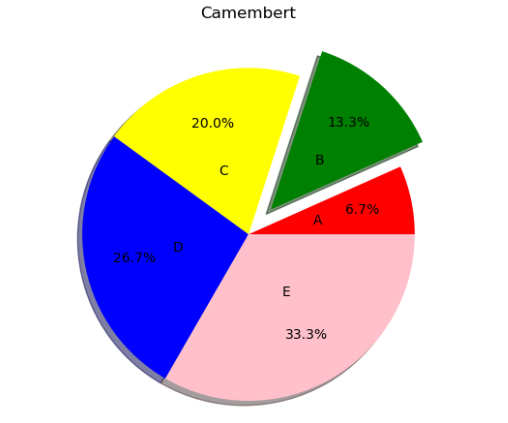
1. **Afficher un histogramme et un camembert**

La fonction "bar" permet d'obtenir un histogramme.





Nous utilisons la fonction "pie" pour implémenter "camembert"



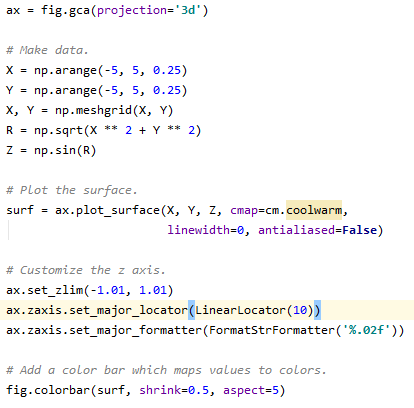
**6. Afficher une surface 2D dans un espace 3D (mesh)**

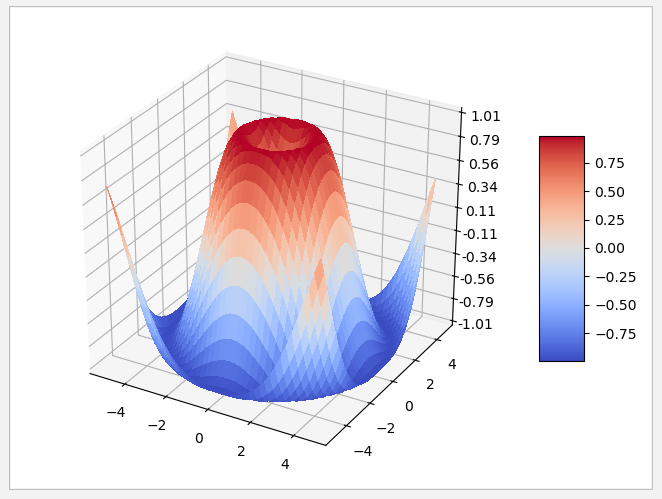
Pour réaliser un dessin en 3D, nous importons la bibliothèque suivante:

from mpl\_toolkits.mplot3d import Axes3D

from matplotlib.ticker import LinearLocator, FormatStrFormatter

Le processus de mise en œuvre est le suivant:





# TP5 Base de données

## Travail du TP

