

Programmation PYTHON

Cours 1

Nassim ZELLAL

2020/2021

Objectifs du cours PYTHON

- Maîtriser la programmation en Python, qui permet d'aborder les notions fondamentales de la programmation pour ce langage, très utilisé en TAL (Traitement Automatique des Langues).
- Manipuler des chaînes de caractères permettant de traiter des problématiques liées à la langue et à l'analyse de fichiers textuels : analyse de données textuelles, analyse statistique de données textuelles, analyse de corpus, génération de concordances, stemming (bibliothèque NLTK), Traitement Automatique des Langues.
- Maîtriser les expressions régulières permettant de manipuler des chaînes de caractères, c'est-à-dire de trouver les portions de la chaîne correspondant à un modèle (patron).
- Savoir mettre en œuvre des solutions, dans les situations les plus fréquemment rencontrées en pratique, pour le traitement informatique des jeux de caractères (conversion et/ou correction de l'encodage d'un fichier).

À quoi sert PYTHON?

- Traitement de données textuelles.
- Manipulation de chaînes de caractères.
- Gestion de fichiers.
- Utilisation du langage des expressions régulières.
- Développement d'applications en TALN/Extraction d'information/Apprentissage automatique.
- Administration système et réseau.
- Manipulation de documents structurés (DOM, SAX).

Interpréteur de Python

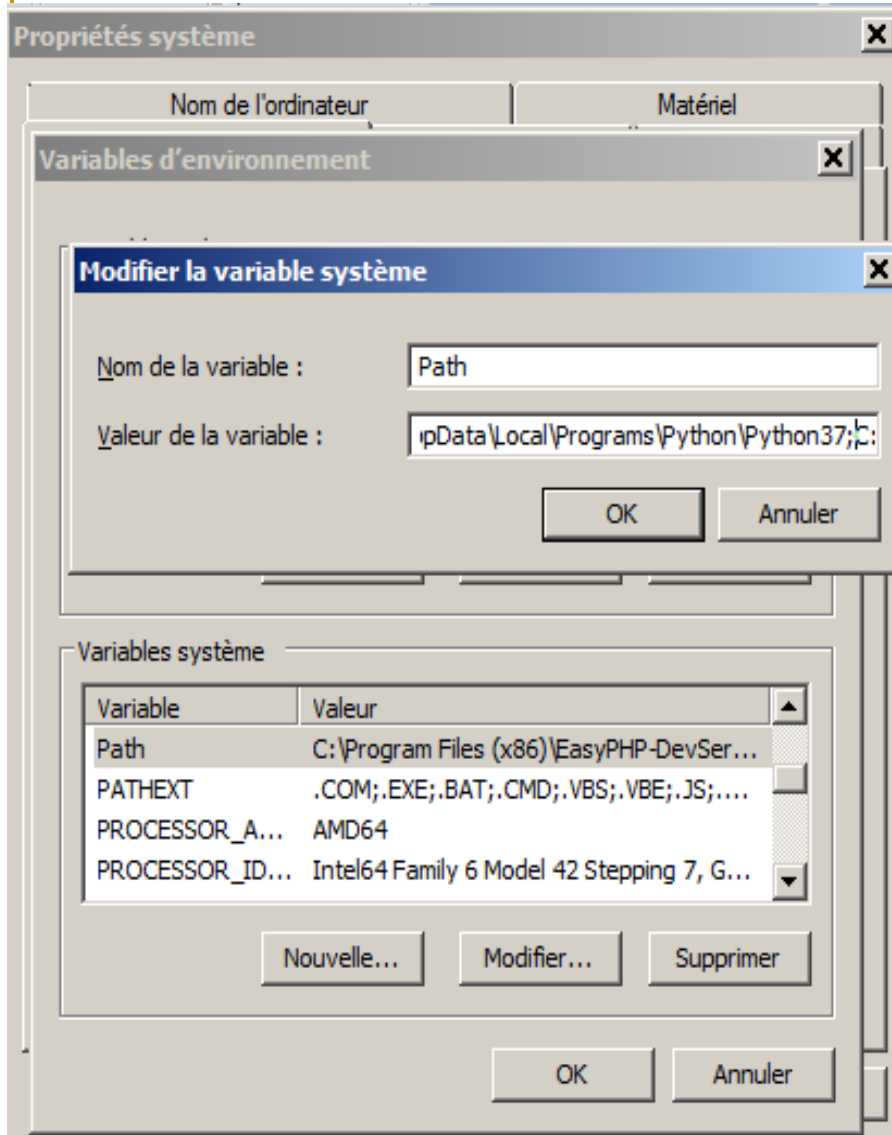
C:\Windows\System32\cmd.exe - python

```
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\user\Desktop>python
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

Fonction **print()** sur
l'interpréteur de Python

Variable d'environnement de Python



Si vous avez le message suivant :

'python' n'est pas reconnu en tant que commande interne, vérifiez votre variable

d'environnement. Aller dans :

Panneau de configuration\Systeme\Paramètres Système avancés\Variables d'environnement\Variables système\Path.

Modifiez la variable "Path", en y ajoutant les chemins de votre Python : e.g.,

C:\xxxxx\xxxxx\AppData\Local\Programs\Python\Python37

C:\xxxxx\xxxxx\AppData\Local\Programs\Python\Python37\Scripts

Mon premier script en Python

nom-du-script.py



1

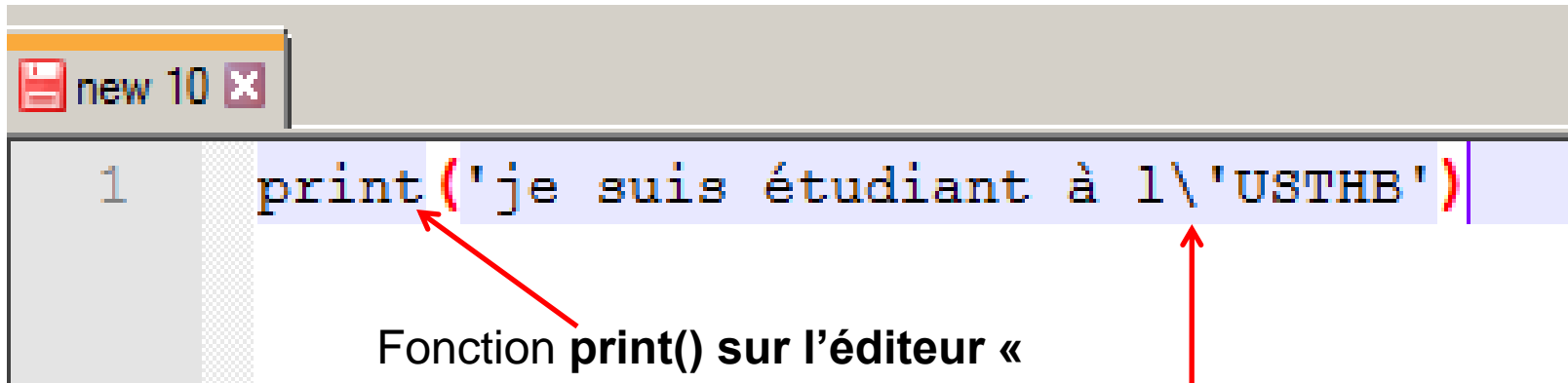
2

1- nom de votre script

2- extension de votre script en .py

Mon premier script en Python

« Notepad++ » - ouverture d'un nouveau fichier



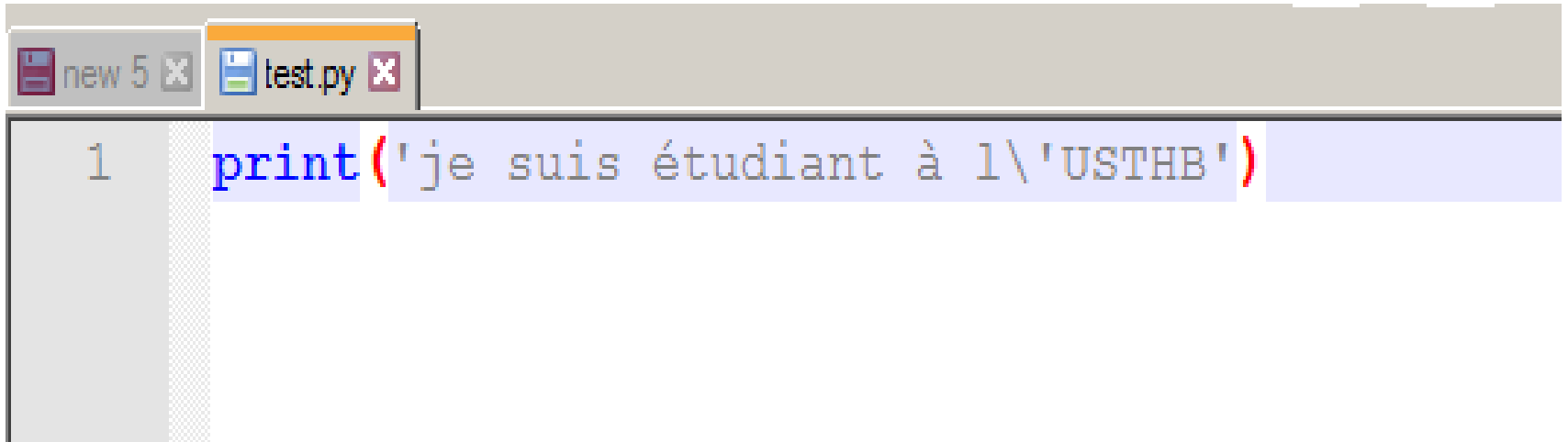
The image shows a screenshot of the Notepad++ text editor. The title bar at the top indicates a new file named 'new 10'. The editor contains a single line of Python code: `print('je suis étudiant à l\'USTHB')`. The code is highlighted in light blue. A red arrow points from the text 'Fonction print() sur l\'éditeur « Notepad++ »' to the `print` function. Another red arrow points from the text 'Il faut déspecialiser le caractère \' avec l\'antislash' to the `\'` character in the string.

```
1 print('je suis étudiant à l\'USTHB')
```

Fonction **print()** sur l'éditeur «
Notepad++ »

Il faut déspecialiser le caractère `'` avec
l'antislash

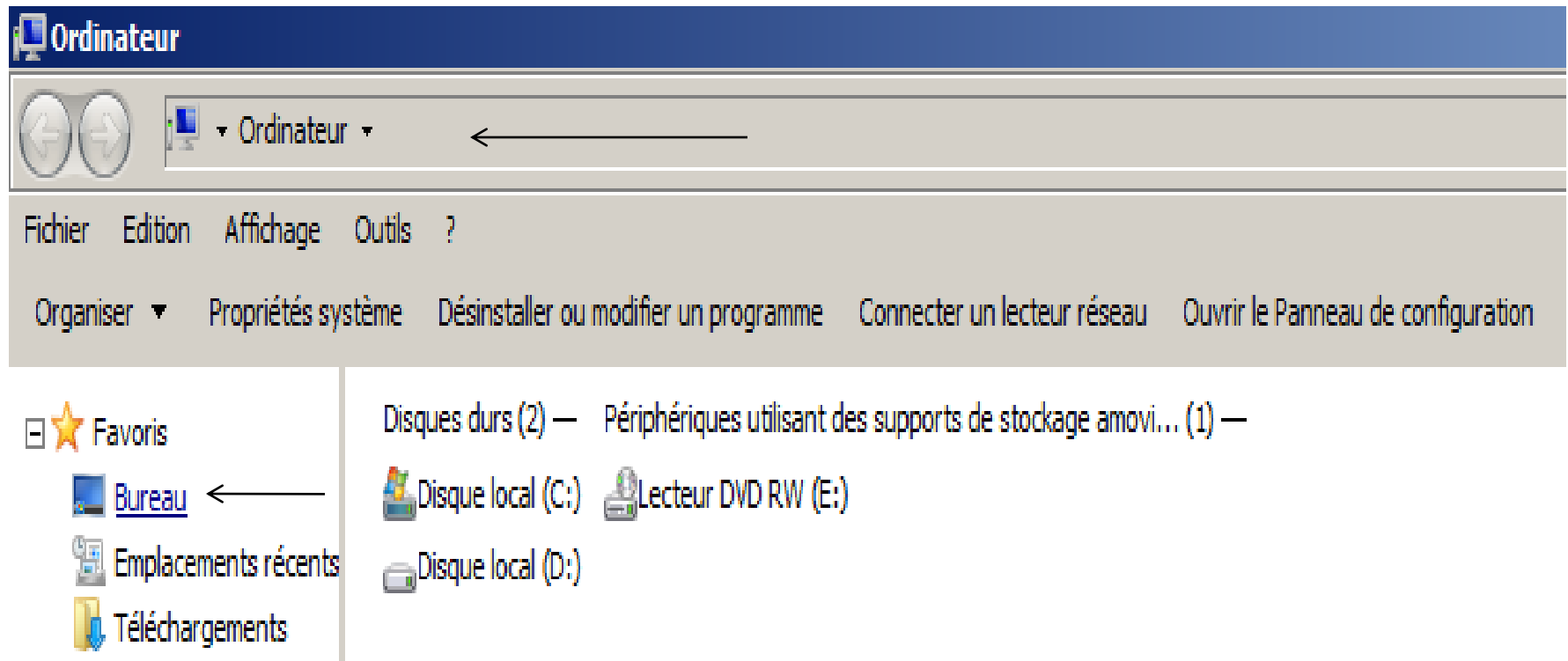
Enregistrement du fichier sur le bureau (Desktop) avec l'extension .py



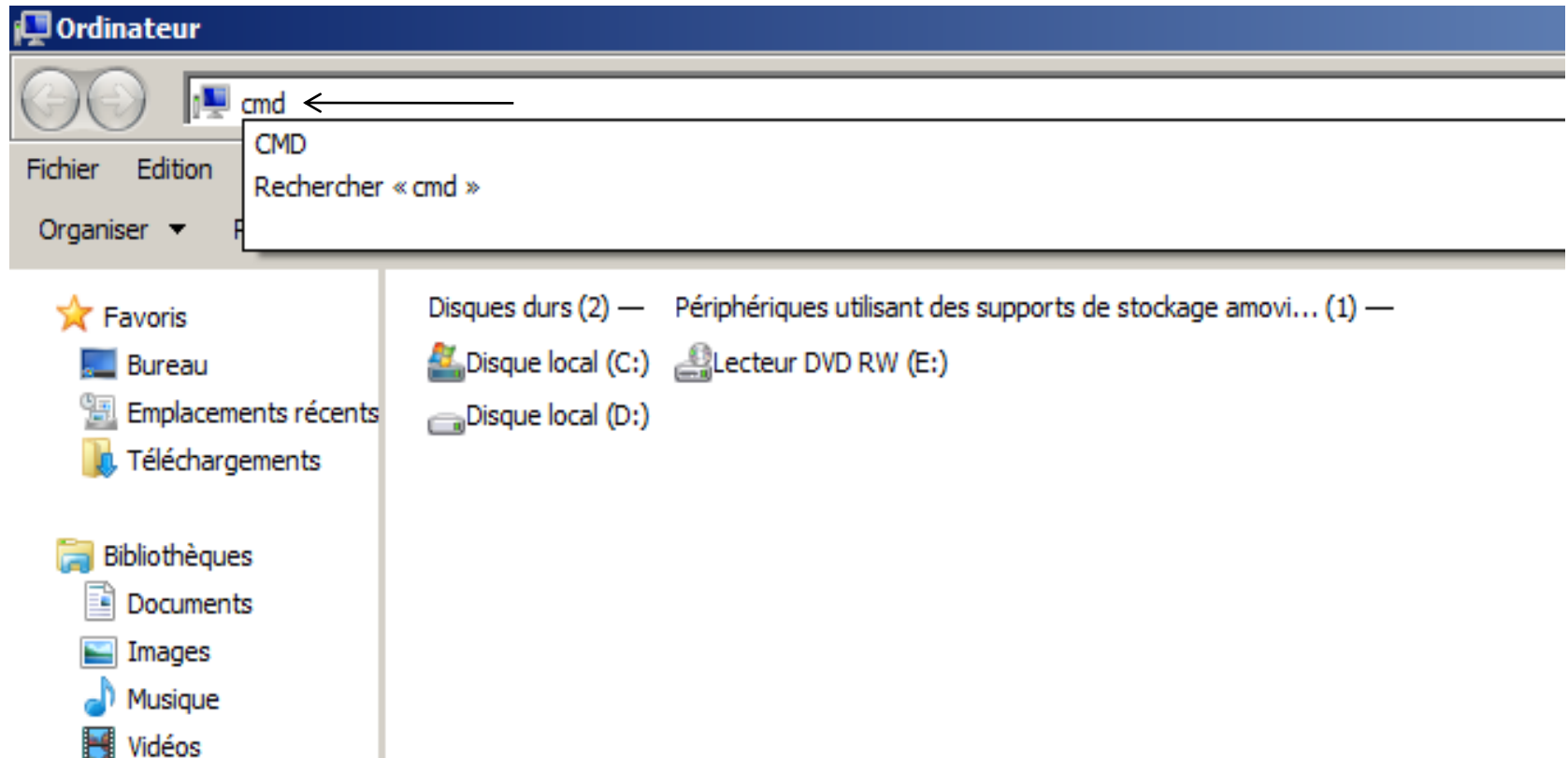
The image shows a screenshot of a code editor window. The title bar at the top contains two tabs: 'new 5' and 'test.py'. The 'test.py' tab is active. The editor area shows a single line of Python code: `print('je suis étudiant à l\'USTHB')`. The line number '1' is visible on the left side of the editor.

```
1 print('je suis étudiant à l\'USTHB')
```


Ouverture d'une invite de commandes à l'aide de la barre de chemins d'accès (barre d'adresse de l'Explorateur Windows)



Barre de chemins d'accès (ouverture d'une invite de commandes au niveau du bureau)



Exécution d'un script Python sur une invite de commandes Windows

```
C:\Users\user\Desktop>_
```

Ouvrir une invite de commandes

Exécution d'un script Python sur une invite de commandes Windows

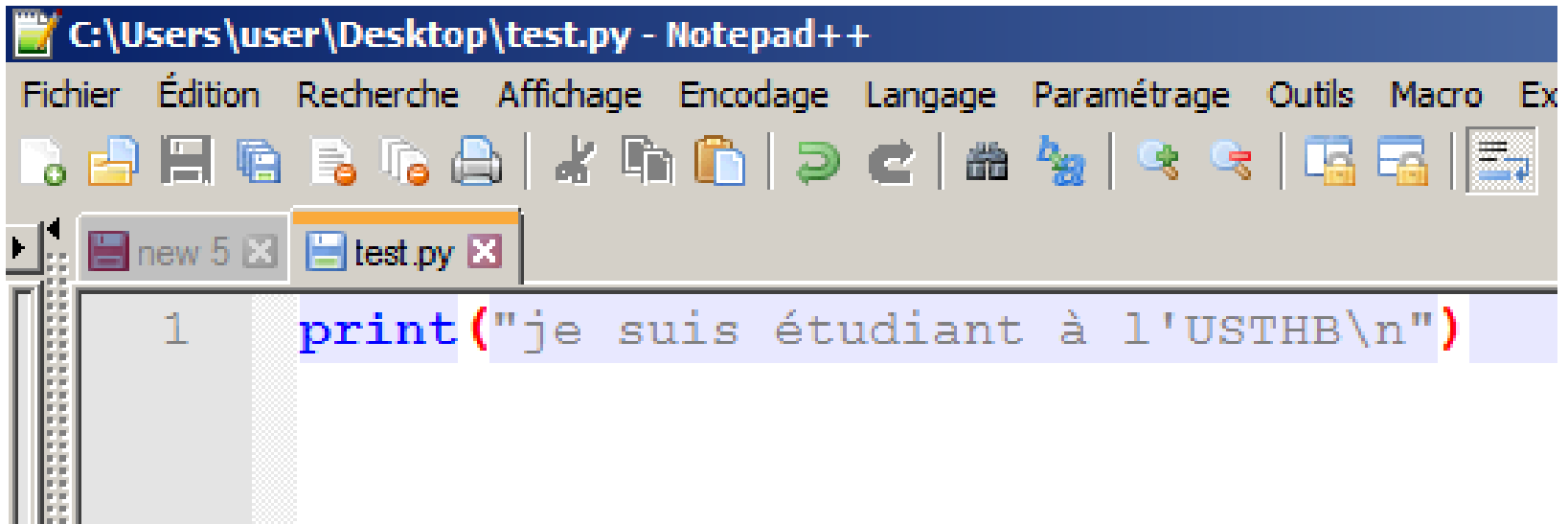
```
C:\Users\user\Desktop>test.py_
```

Tapez le nom de votre script pour l'exécuter

Exécution d'un script Python sur une invite de commandes Windows

```
C:\Users\user\Desktop>test.py  
je suis étudiant à l'USTHB
```

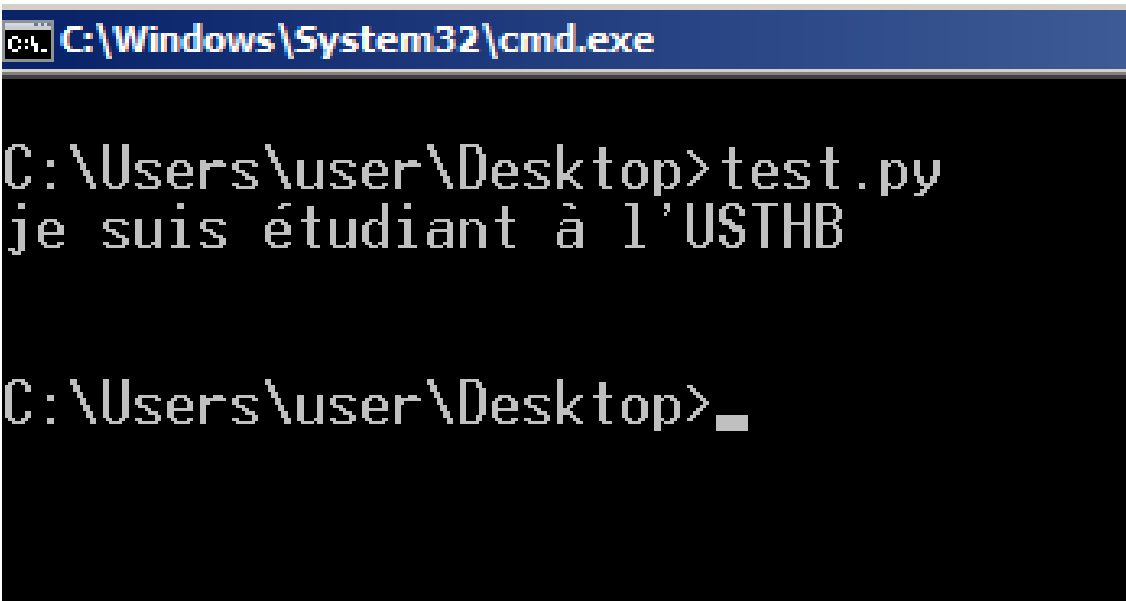
Saut de ligne avec « \n » en Python



The image shows a screenshot of the Notepad++ application window. The title bar reads "C:\Users\user\Desktop\test.py - Notepad++". The menu bar includes "Fichier", "Édition", "Recherche", "Affichage", "Encodage", "Langage", "Paramétrage", "Outils", "Macro", and "Ex". The toolbar contains various icons for file operations, editing, and development. The tab bar shows two tabs: "new 5" and "test.py". The main text area displays a single line of Python code: `1 print("je suis étudiant à l'USTHB\n")`. The code is syntax-highlighted, with "print" in blue, the string in red, and the newline character "\n" in black. The line number "1" is visible in the left margin.

```
1 print("je suis étudiant à l'USTHB\n")
```

Ajout d'un saut de ligne « \n »



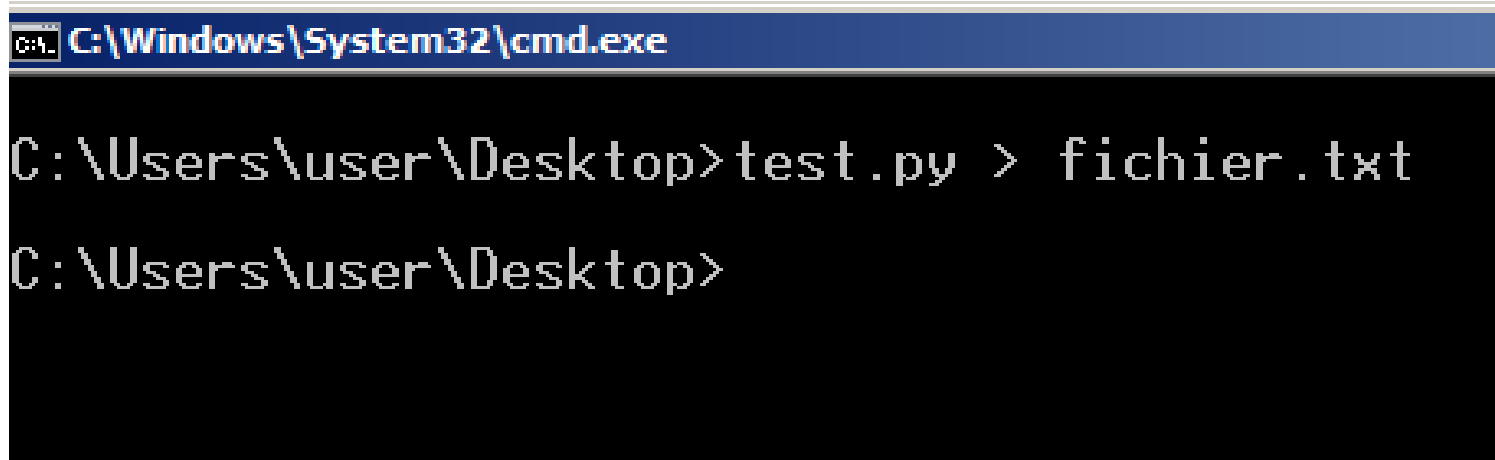
```
C:\Windows\System32\cmd.exe

C:\Users\user\Desktop>test.py
je suis étudiant à l'USTHB

C:\Users\user\Desktop>_
```

The image shows a Windows command prompt window with a blue title bar that reads "C:\Windows\System32\cmd.exe". The command prompt is black with white text. It shows the user running a command "test.py" from the directory "C:\Users\user\Desktop". The output of the command is "je suis étudiant à l'USTHB", which is displayed on a new line. The prompt then returns to "C:\Users\user\Desktop>_" with a cursor.

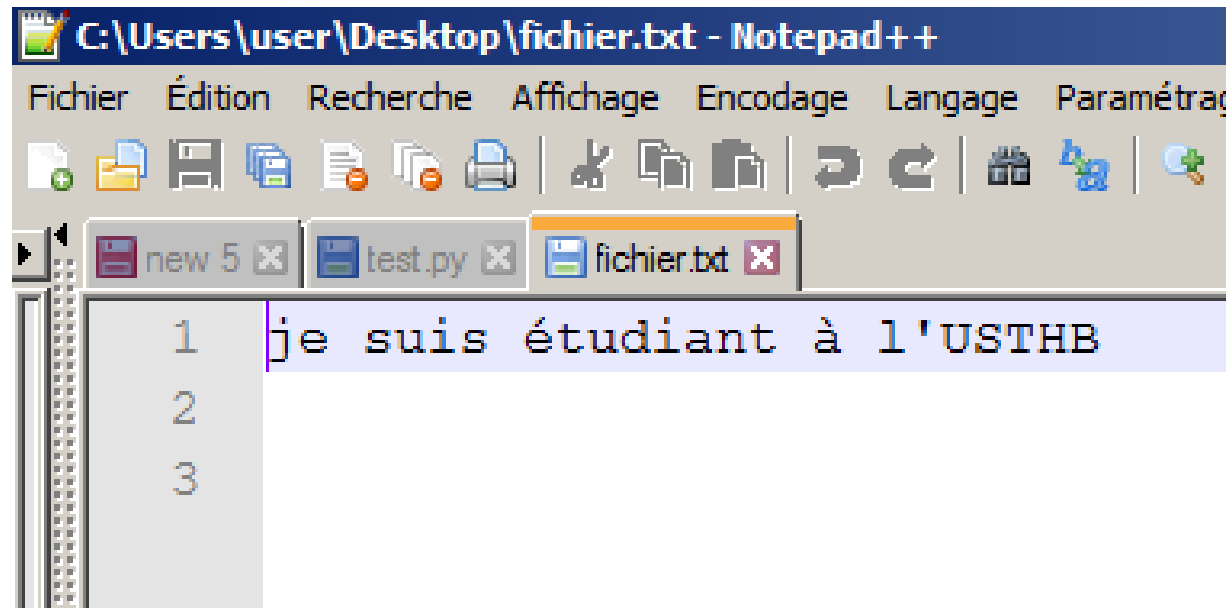
Rediriger le résultat de la commande print vers un fichier



```
C:\Windows\System32\cmd.exe  
C:\Users\user\Desktop>test.py > fichier.txt  
C:\Users\user\Desktop>
```

The image shows a screenshot of a Windows command prompt window. The title bar at the top reads "C:\Windows\System32\cmd.exe". The command prompt shows the current directory as "C:\Users\user\Desktop". The user has entered the command "test.py > fichier.txt", which has been executed. The prompt now shows "C:\Users\user\Desktop>".

Résultat de la redirection



Types d'objets en Python - Classes

- **Classe str (String) : chaînes de caractères.**
- **Classe int (Integer) : nombres entiers.**
- **Classe float : nombres flottants (ou nombres réels).**
- **Remarque : Toute variable est un objet.**
- **Remarque : Toute fonction/méthode est un objet.**
- **Remarque : Un objet est une instance d'une classe.**
- **Donc, en Python, tout est objet.**

Classes « Str », « Int » et « Float » (types de données)

- **a=5**
 - **s=str(a) # fonction str(). On l'appelle aussi « constructeur ».**
 - **print(s)**
 - -----
 - **b="5"**
 - **i=int(b) #fonction int(). On l'appelle aussi « constructeur ».**
 - **print(i)**
 - -----
 - **c=3.2**
 - **f=float(c) #fonction float()#avec int() on perd la précision**
 - **print(f). float() est également un « constructeur » de nombres à virgule flottante.**
-
- **Remarque importante : Tout type de donnée est une classe.**

Nommage des variables en Python

- Lettres (minuscules ou majuscules - ASCII : **American Standard Code for Information Interchange**).
- _ (tiret bas / underscore)
- Exemples :
 - maChaine
 - MaChaine2
 - Ma_2eme_Chaine
 - 2eme_chaine

Opérateurs d'affectation (assignment)

- Opérateur d'affectation simple :
 - `a=6`
 - `b=2`
 - `c=b`
 - `#-----#`
 - `a="il"`
 - `b="écrit"`
 - `c=b`
- Opérateurs d'affectation composés :
 - `a="il "`
 - `b="écrit"`
 - **`a=a+b #a+=b`**
 - **`print(a)`**
 - **`> il écrit`**

Exemple d'affectation de deux variables Str

- `chaine_1="Spécialité"`
- `chaine_2="Académique"`
- `print("chaine_1 chaine_2")`

Résultat de l'affectation

C:\Windows\System32\cmd.exe

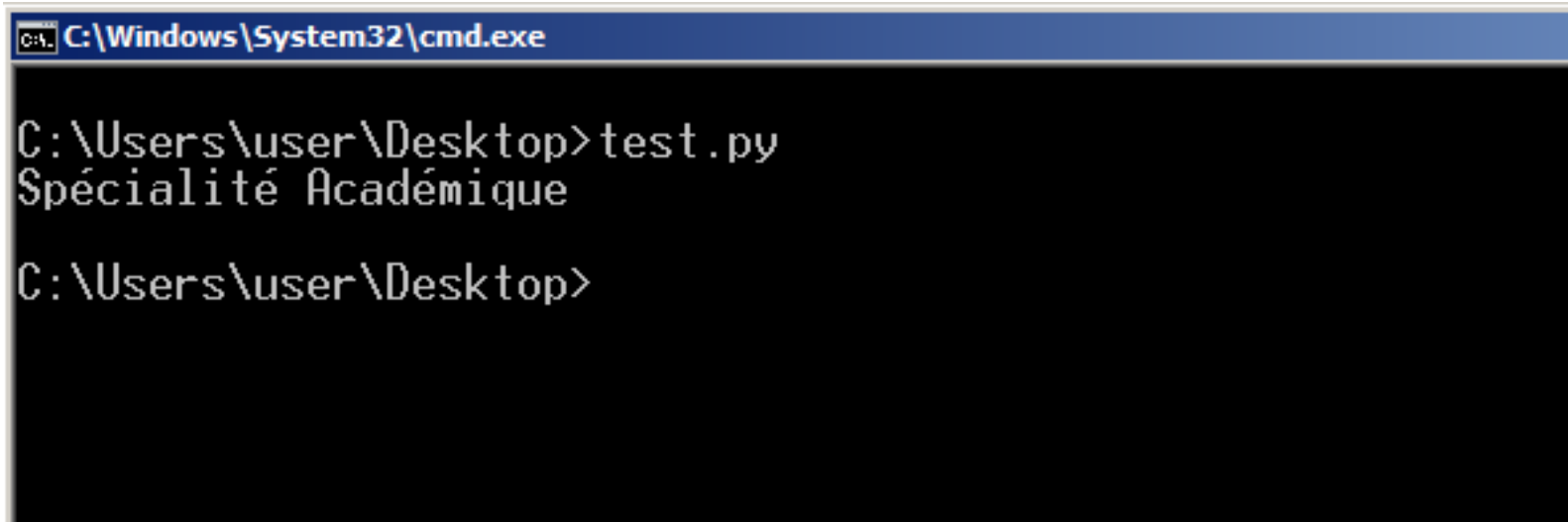
```
C:\Users\user\Desktop>test.py  
chaine_1 chaine_2
```

```
C:\Users\user\Desktop>
```

Exemple d'affectation de deux variables Str

- `chaine_1="Spécialité"`
- `chaine_2="Académique"`
- `print(chaine_1+" "+chaine_2)`#concaténation de chaînes de caractères avec le +
- `> Spécialité Académique`

Résultat de l'affectation



```
C:\Windows\System32\cmd.exe  
  
C:\Users\user\Desktop>test.py  
Spécialité Académique  
  
C:\Users\user\Desktop>
```

The image shows a screenshot of a Windows command prompt window. The title bar at the top reads "C:\Windows\System32\cmd.exe". The command prompt shows the user's current directory as "C:\Users\user\Desktop". The user has entered the command "test.py", which has been executed, resulting in the output "Spécialité Académique". The prompt is now ready for the next command.

Types d'objets en Python - suite

- **Listes - classe « list »**
- **Tuples - classe « tuple »**
- **Dictionnaires - classe « dict »**

Les listes


Une liste est un objet pouvant contenir d'autres objets (entiers, chaînes de caractères). Vous pouvez accéder à chaque élément de cette liste à travers son indice.

- `maliste= ["dog", "cat", "mouse"]` #une liste avec 3 objets
- `maliste = [23, 45, 300, 98]`
- `maliste[2]` #imprime mouse
- `maliste[1]` #imprime 45
- `len(montab)-1` #**imprime le dernier indice du tableau avec la fonction len()**
- `maliste[-1]` #**imprime le dernier élément du tableau**

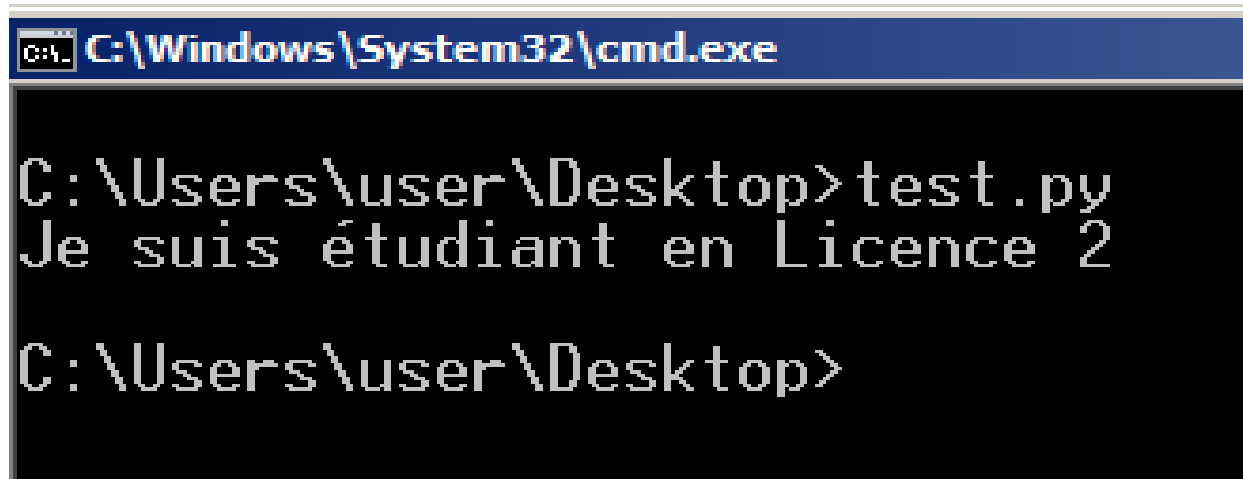
Exemple d'affectation d'une liste

```
maList=["Licence",2]  
# Ceci est un commentaire  
print("Je suis étudiant en "+maList[0]+" "+str(maList[1]))
```

Commentaire en Python



Résultat de l'affectation



```
C:\Windows\System32\cmd.exe  
  
C:\Users\user\Desktop>test.py  
Je suis étudiant en Licence 2  
  
C:\Users\user\Desktop>
```

A screenshot of a Windows command prompt window. The title bar at the top is blue and contains the text "C:\Windows\System32\cmd.exe". The main area of the window is black with white text. It shows a command prompt session where the user has run a script named "test.py" from the directory "C:\Users\user\Desktop". The script's output is "Je suis étudiant en Licence 2". The prompt is currently at "C:\Users\user\Desktop>".

Création d'une liste vide

- `maliste = list()` # création d'une liste vide grâce au « constructeur » `list()`, permettant donc de créer une liste.
- `print(type(maliste))` # la fonction `type()` donne le type de donnée
- `> <classe 'list'>`
- `maliste = []` # création d'une liste vide
- `print(type(maliste))`
- `> <classe 'list'>`

L'opérateur « del »

- **Del** : supprime un élément d'une liste.

```
tab=[5,6,3]
```

```
del tab[2]
```

```
print (tab)
```

```
> [5,6]
```

Types d'objets en Python - suite

- **Listes - classe « list »**
 - **Tuples - classe « tuple »**
 - **Dictionnaires - classe « dict »**
-

Les tuples

- Les tuples sont des listes immuables, qu'il est impossible de modifier.
- Pour créer un tuple, on utilise des parenthèses au lieu des crochets.
- À quoi sert un tuple ?
- Affectation multiple de variables.
- Même sans mettre de parenthèses, Python comprend qu'il doit créer un tuple pour faire l'affectation.
- Voir le slide suivant ...

Les tuples

- Exemple :
- `a = ()` #tuple vide
- `b= (1, 2, 3)` #tuple contenant trois entiers
- `(a,b)=(3,5)` #**affectation multiple**
- `t=(a,b)`
- `print(type(t))` #tuple
- `a,b=3,5` #**affectation multiple**
- `t=(a,b)`
- `print(type(t))` #tuple
- `a=3,5`
- `print(type(a))` #tuple
- **#L'opérateur "del" ne fonctionne pas sur les tuples.**

Méthode « split() » - listes

```
a="Bonjour j'aime l'USTHB"
```

```
mots=a.split()#l'espace est le délimiteur par défaut
```

```
print(type(mots)) #mots est une liste (classe « list »)
```

```
#-----#
```

```
x = "bleu,rouge,vert"
```

```
(a,b,c)=x.split(",")#on utilise la virgule comme délimiteur
```

```
print(a+" comme le ciel "+b+" comme la tomate "+c+"  
comme une plante")
```

```
t=(a,b,c)
```

```
print(type(t))
```

- Dans l'exemple ci-dessus, **a**, **b** ou **c** pris séparément appartiennent à la classe « str », mais **t** appartient à la classe « tuple ».

Passage d'arguments avec la variable spéciale de type « list » `argv[x]` et le module « sys »

import sys #importer le module « sys »

```
print("Je suis étudiant en "+sys.argv[1]+" et je  
m'appelle "+sys.argv[2])
```

Ouvrir une ligne de commandes et tapez ceci :

```
>test.py informatique un_prenom
```

```
#-----#
```

On peut également faire :

```
from sys import argv
```

```
print("Je suis étudiant en "+argv[1]+" et je m'appelle  
"+argv[2])
```

```
>test.py informatique un_prenom
```

Résultat de la récupération des deux arguments de type « Str » sur l'invite (ligne) de commandes

```
C:\Users\user\Desktop\test>test.py informatique un_prenom  
Je suis étudiant en informatique et je m'appelle un_prenom
```

Exercice 1 - quelle est la ligne de commandes à écrire?

- `import sys`
- `mark1 = sys.argv[1]`
- `mark2 = sys.argv[2]`
- `mark3 =sys.argv[3]`
- `course1 = sys.argv[4]`
- `course2 = sys.argv[5]`
- `course3 = sys.argv[6]`
- `print("Cours1\t"+course1+" = note1\t"+mark1+"\n")`
- `print("Cours2\t"+course2+" = note2\t"+mark2+"\n")`
- `print("Cours3\t"+course3+" = note3\t"+mark3+"\n")`
- `print("-----\n")`
- `resultat=int(mark1) + int(mark2) + int(mark3)`
- `resultat=resultat/3`
- `print("moyenne\t="+str(resultat))`

Exercice 2

- Écrire un script Python permettant de diviser la phrase « **J'aime l'USTHB** » passée en argument, en utilisant comme caractère de division l'apostrophe.
- Le mot « **USTHB** » doit être affiché sur votre invite de commandes.

Adresse de téléchargement de PYTHON

<https://www.python.org/downloads/>

Mon courriel

zellal.nassim@gmail.com
