

Introduction à la programmation Python

■ ■ ■ Entrées clavier et Répétitives

- 1 – Écrire un programme :
 - a. demandant à l'utilisateur d'entrer une valeur entière ;
 - b. affichant sur une même ligne autant de « * » que la valeur entrée précédemment.
- 2 – Écrire un programme qui :
 - ▷ demande à l'utilisateur :
 - ◊ son numéro de sécurité sociale sur 13 chiffres ;
 - ◊ la clé de vérification ;
 - ▷ décompose et affiche la valeur de chaque champs, indiqué en nombre de chiffres :
 - ◊ un pour le sexe ;
 - ◊ deux pour l'année de naissance ;
 - ◊ deux pour le mois de naissance ;
 - ◊ deux pour le département de naissance ;
 - ◊ trois pour la commune de naissance (aucun département ne comporte plus de 999 communes) ;
 - ◊ trois pour un numéro d'ordre dans le mois de naissance ;
 - ▷ indique si le numéro saisi et la clé correspondent.

Le calcul de la clé de vérification du numéro de sécurité sociale :
 $clé = 97 - (numéro \bmod 97)$

- 3 – Écrire un programme de devinette :
 - a. le programme choisit une valeur aléatoire entre 1 et 100 ;
 - b. l'utilisateur entre la valeur qu'il pense avoir deviné ;
 - c. le programme lui répond :
 - ◊ si la valeur entrée est supérieure ou inférieure à la valeur à deviner ;
 - ◊ si la valeur entrée est celle choisie par le programme, le nombre d'entrées réalisées pour la découvrir ;
 - d. le programme continue tant que l'utilisateur n'a pas deviné la valeur choisie par le programme.

■ ■ ■ Gestion des chaînes de caractère

- 4 – Écrire un programme permettant de vérifier que le numéro d'une carte de crédit est correct :

4	1	3	7	8	9	4	7	1	1	7	5	5	9	0	4
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

On double chaque valeur de rang impair :

8	1	6	7	16	9	8	7	2	1	14	5	10	9	0	4
1		3		5		7		9		11		13		15	
				1+6=7						1+4=5		1+0=1			

Si la valeur est sur 2 chiffres on fait la somme de ces deux chiffres.

D'où le résultat final :

8	1	6	7	7	9	8	7	2	1	5	5	1	9	0	4
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

On fait la **somme** de tous les chiffres obtenus : $8 + 1 + 6 + 7 + 7 + 9 + 8 + 7 + 2 + 1 + 5 + 5 + 1 + 9 + 0 + 4 = 80$, si le **dernier chiffre** est 0 $\Rightarrow$ le numéro de la carte est **valide**, sinon le numéro est invalide.

- 5 – Écrire un programme affichant pour chaque caractère d'une chaîne, son rang :
 - a. en hexadécimal ;
 - b. en binaire ;
 Vérifiez comment faire la gestion des caractères accentués.

■ ■ ■ Gestion des listes modifiables

- 6 – Écrire un programme qui réalise l'insertion d'une liste d'éléments dans une liste existante à un emplacement donné par son indice.
- 7 – Soit le code suivant :

```
#!/usr/bin/python3

import sys
import subprocess

resultat = subprocess.run('ls *.py', shell=True, stdout=subprocess.PIPE)
liste_fichiers = resultat.stdout.splitlines()

print (liste_fichiers)
```

- a. Utilisez le pour « écrire un programme prenant la liste des fichiers contenus dans un répertoire, et qui ouvre et affiche la première ligne de chacun de ces fichiers ».
- b. En utilisant la commande `ls -F`, affichez la liste des répertoires, puis des exécutables et enfin la liste des fichiers.

■ ■ ■ Manipulation de fichiers

- 8 – Écrire un programme qui compte le nombre de lignes d'un fichier.
- 9 – Écrire un programme qui ouvre un premier fichier et crée un nouveau fichier contenant une ligne sur deux du premier fichier.

■ ■ ■ Gestion des dictionnaires & opérations d'éclatement et de recomposition (split & join)

- 10 – Écrire deux programmes :
 - a. le premier qui effectue l'écriture de données dans un fichier texte (organisées sous forme de lignes de textes, où chaque ligne est composée de caractères et terminée par un `\n`). Ces informations sont le *nom*, *prénoms* et *adresse* d'un individu. Elles sont saisies au clavier et enregistrées dans le fichier « `carnet_adresse.txt` » :
 - une ligne par individu,
 - chaque ligne composée de la manière suivante : `<nom> : <prénoms> : <adresse>`Exemple : `Toto : Titi : 123 av Albert Thomas`
 - b. le second qui lit le fichier créé par le premier et affiche les informations *nom*, *prénom* et *adresse*, chacune sur une ligne, précédée de l'intitulé du champs (vous séparerez les infos de chaque individu par une ligne contenant des #).
- 11 – Écrire un programme qui effectue la lecture et la construction d'un dictionnaire à partir du contenu d'un fichier d'options structuré de la façon suivante :

```
nom_option = valeur_option
nom_option_2 = valeur_option_2
...
```

Afficher la liste des associations (options, valeur).

■ ■ ■ Expressions régulières

12 – Gestion d'un compte bancaire à l'aide d'un simple fichier au format texte permettant de :

- ◇ faire des opérations de débit ;
- ◇ faire des opérations de crédit ;
- ◇ définir le solde initial du compte ;
- ◇ calculer et afficher le solde courant du compte.

Exemple de fichier à traiter :

```
1:1500,00
2+345,00 # virement compte epargne
3-450,00 # reparation voiture
4-80,50
5-----
6-25,80 # repas
7-78,60 # livres
8-67,90 # imprévu
9-----
```

Explications : la ligne commençant par un :

★ « : » permet de définir le solde initial ;

★ « + » expriment un crédit ;

★ « - » expriment un débit ;

Les lignes contenant « ----- » calculent et affichent le solde du compte en tenant compte uniquement des opérations notées précédemment.

Les commentaires sur la nature des opérations sont indiqués du caractère « # » jusqu'à la fin de la ligne.

Écrire un programme réalisant le travail et fournissant l'affichage suivant :

```
xterm
Solde initial : 1500
+345,00 # virement compte epargne
-450,00 # reparation voiture
-80,50
-----
= 1314,5
-25,80 # repas
-78,60 # livres
-67,90 # imprévu
-----
=1142,2
```