ISN – Informatique et Sciences du Numérique

TP2 PYTHON : SAISIR DES DONNEES

1 – INSTRUCTION « INPUT »

L'instruction « `input` » permet d'obtenir des données de l'utilisateur. Cette instruction ***interrompt*** *le programme,* ***affiche un message*** *dans une boîte de dialogue et* ***attend*** *que l'utilisateur* ***saisisse des caractères****. La saisie est validée par l'appui sur le bouton « OK ».*

Exercice 1

1. **Taper** dans l'éditeur de scripts les lignes suivantes :

```
deplacement = input(" Quelle direction ?")
print(deplacement)
```

2. **Enregistrer** puis **exécuter** le script. **Tester-le** en entrant une des chaines suivantes : "Haut", "Bas", "Droite" ou "Gauche".

2 – AFFICHER UNE CHAINE DE CARACTERES

Exercice 2

1. **Taper** dans l'éditeur de scripts les lignes suivantes :

```
Nombre = input("Entrer un nombre entier :")
Calcul = Nombre + 2
print("Le résultat est :",Calcul)
```

2. **Enregistrer** puis **exécuter** le script. **Tester-le**. **Justifier** pourquoi le programme génère une erreur.

3. **Modifier** le script de la manière suivante :

```
Valeur = input("Entrer un nombre entier :")
Nombre = int(Valeur)
Calcul = Nombre + 2
print("Le résultat est :",Calcul)
```

4. **Enregistrer** puis **exécuter** le script. **Tester-le**.

5. **Donner** le rôle de l'instruction « `int` ». **Indiquer** à quel type appartiennent les variables « Valeur », « Nombre » et « Calcul ».

6. **Modifier** le script de la manière suivante :

```
Nombre = int(input("Entrer un nombre entier :"))
Calcul = Nombre + 2
print("Le résultat est :",Calcul)
```

7. **Enregistrer** puis **exécuter** le script. **Tester-le**.

8. **Indiquer** à quel type appartiennent les variables « Nombre » et « Calcul ».

3 – FONCTIONS

*Une fonction est une **portion de code** (sorte de sous-programme) que l'on peut appeler au besoin. L'utilisation des fonctions permet : **d'éviter la répétition** de lignes de code identiques, la **réutilisation dans d'autres scripts** par l'intermédiaire du mécanisme de l'import ; de **décomposer une tâche complexe** en tâches plus simples. On obtient ainsi des **programmes plus courts et plus lisibles**. Une fonction est définie au moyen de l'instruction « `input` ». La syntaxe est la suivante :*

Syntaxe

```
def nomFonction(parametres1,parametre2,parametre3):
    """
       Documentation de la fonction
    """
    Bloc<instructions>
    return resultat
```

Exercice 3

1. **Taper**, dans l'interpréteur, les lignes suivantes qui définissent la fonction « carre » qui permet le calcul du carré de tout nombre x :

```
def carre(x):
    """
       La fonction carre permet le calcul du carré de tout nombre x
       x : nombre dont il faut calculer le carré
       valeur de retour : carré de x
    """
    Val = x ** 2
    return Val
```

2. **Enregistrer** puis **exécuter** le script. **Tester-le** en tapant dans l'interpréteur `>>> carre(4)` par exemple. **Tester** le fonctionnement pout d'autres valeurs de x.

3. **Taper** dans l'interpréteur `>>> help(carre)`. **Donner** le rôle de la « Documentation de la fonction » dans la définition de la fonction.

4. **Modifier** le script précédant afin que le nombre x soit entré au clavier par l'utilisateur.

4 – COMMENTAIRES

*Un commentaire est un **court texte indiquant ce que fait une partie du code**. Il ne représente aucune instruction et ne sera donc pas interprété. Ces commentaires sont utiles aux développeurs ou à d'autres personnes voulant comprendre rapidement le code. Pour être utile, un commentaire doit être pertinent et concis.*

*Un **commentaire en ligne** commence par le symbole « # ».*

Exemples

```
Nombre = int(Valeur)    # Conversion d'un caractère en entier

#--------------------------
# Jeu du labyrinthe
#--------------------------
```

*Il est possible également de commenter un code à l'aide de chaînes de caractères appelées **docstring**. Ce type de commentaires est plutôt dédié à la documentation des fonctions. Il sera accessible à l'utilisateur au moyen de l'instruction « help ».*

Exemple

```
def carre(x):
    """
        La fonction carre permet le calcul du carré de tout nombre x
        x : nombre dont il faut calculer le carré
        valeur de retour : carré de x
    """
```

Exercice 4

1. **Ouvrir** le script lab1.py édité lors du TP1.
2. **Ajouter** des commentaires.

5 – LABYRINTHE

Exercice 5 : Labyrinthe

1. **Ouvrir** le script lab1.py édité lors du TP1. **Enregistrer**-le sous le nom lab2.py.
2. **Créer** une fonction « `affiche_bordure(taille)` » permettant l'affichage uniquement de la bordure du labyrinthe dont les dimensions seront fixées par la variable taille.

3. **Créer** une fonction « `affiche_labyrinthe(lab)` » permettant l'affichage du labyrinthe (lors de l'appel de la fonction la variable « lab » correspondra à la variable « niveau_1 »).
4. **Créer** une fonction « `choix_joueur()` » permettant de demander au joueur et de retourner quel déplacement (Haut, Bas, Gauche, Droite) doit être effectué.
5. **Editer** le programme principal afin que :
   - Les bordures du labyrinthe soient affichées
   - Le labyrinthe soit affiché
   - Le déplacement soit demandé au joueur puis affiché.
6. **Tester** le fonctionnement du programme et des différentes fonctions.