Apprentissage de la programmation

Michel Blancard
p7@michelblancard.fr

Objectifs pédagogiques

A la fin de ce cours :

- vous saurez ce qu’est la programmation

- vous ne saurez pas *bien* programmer, mais vous saurez dans quelle direction aller pour *améliorer* votre pratique

- vous aurez un vernis sur quelques enjeux sociétaux relatifs au numérique

Structure du cours

Pour chaque séance :

1. Exposé d’un groupe (10-20mn)

2. Introduction sur un thème technique ou un enjeux du numérique (20-30mn)

3. Travaux sur machines

Plan des séances

1. Structures de données (technique)
2. Modèles économiques du numérique
3. Informatique et démocratie
4. Intelligence artificielle
5. Droit du numérique
6. Contrôle de version avec Git (technique)
7. Informatique en entreprise
8. Méthodes de développement logiciel
9. Métiers de l’informatique
10. (à définir)

Evaluation

(à définir d’ici 2-3 semaines)

- 25 % assiduité
- 25 % exposé
- 25 % résultat des TD
- 25 % participation aux TD
- (25 % contrôle de connaissance)

Structures de données

1. Nombre
2. Chaîne de caractères
3. Dictionnaire
4. Liste
5. Tuple
6. None

1. Les nombres

Exemples :

42
-5
0
65183421568423213586543213598231586468
6.0487
True
False

1. Les nombres

Ce qu’on peut faire avec :

42 * 54
-5 + 12
8.3 / 5.8

True and False
True or False

2. Chaînes de caractères

Exemples :

"Ce texte"
'Celui là aussi'

2. Chaînes de caractères

Ce qu’on peut faire avec :

"Ce texte" + 'Celui là aussi'

3. Dictionnaire

Exemple 1 :

```
{
   "vert": "Couleur d’un arbre" ,
   "rouge" : "Couleur d’un rubis",
   "bleu" : "Couleur du ciel",
}
```

3. Dictionnaire

Exemple 2 :

```
{
   8: "Lever" ,
   9 : "Cours de botanique",
   12 : "Déjeuner",
   14 : "Cours de programmation",
   18 : "RDV chez le dentiste",
}
```

3. Dictionnaire

Exemple 3 :

```
{
   "nom": "Poirot" ,
   "prénom": "Hercule",
   "profession": "détective",
   "nationalité": "Belge",
   "age" : 45,
}
```

3. Dictionnaire

Ce qu’on peut faire avec :

personnage["age"] = 46
profession = personnage["profession"]

4. Liste

Exemples :

[1, 1, 2, 3, 5, 8]
["un", "deux", "trois"]
[1, 2, "trois"]

4. Liste

Ce qu’on peut faire avec :

```
ma_liste[3] = 5
troisieme_element = ma_liste[3]
len(ma_liste)
```

5. Tuple

C’est une liste qu’on ne peut pas modifier.

Exemples :

(1, 1, 2, 3, 5, 8)
("un", "deux", "trois")
(1, 2, "trois")

5. Tuple

Ce qu’on peut faire avec :

```
troisieme_element = mon_tuple[3]
len(mon_tuple)

mon_tuple[3] = 5  # Non !
```

6. None

Quand une procedure ne renvoie aucun résultat, elle renvoie la valeur spéciale ‘None’.

On ne peut rien faire avec !

Résumé

```
{
   (8, 30) : {
      "lieu": "chez soi",
      "commentaire": "Lever",
   },
   (9, 0) : {
      "lieu": "Paris Diderot",
      "commentaire": "Cours de botanique",
   },
   (18, 15) : {
      "lieu": "5 rue Saint-Jacques, Paris",
      "commentaire": "RDV chez le dentiste",
   },
}
```

Coder en python

1. Les fonctions
2. Les tests
3. Les boucles

1. Les fonctions

```
def ma_fonction(param) :
    variable = param + 3
    return param


resultat = ma_fonction(param=5)
```

2. Les tests

```
ma_variable = 5

if ma_variable + 12 < 54:
    print("plus petit")
else :
    print("plus grand")
```

3. Les boucles

```
for compteur in range(10):
    print(compteur)
```

```
liste_invites = ["Marcel", "Hippolyte", "Berthe"]
for invite in liste_invites:
    print("Bienvenu " + invite)
```

RTFM (Read the f*** manual)

- Référence python : https://docs.python.org/fr/3/

- Google

- Stack Overflow

- Lire les erreurs !