

Jeu de dés

Constructions élémentaires

Christophe Viroulaud

Première - NSI

Pour construire un programme informatique, le langage *assembleur* peut s'avérer rapidement fastidieux.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés
Interpréteur
Variables
Entrée/Sortie
Comparaison
Répétition
Boucle non bornée
Boucle bornée
Bibliothèque

```

1      MOV R0, #tab
2      LDR R1, longueur
3      MOV R2, #0
4      MOV R4, #-1
5  boucle:
6      LDR R3, [R0]
7      CMP R3, R4
8      BGT maxi

```

Code 1 – Extrait d'un programme assembleur

Il existe des **langages de haut-niveau** qui facilitent l'écriture des programmes :

- ▶ Python,
- ▶ C, C++,
- ▶ Java,
- ▶ Javascript
- ▶ ...

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Quelles constructions élémentaires sont suffisantes pour
écrire n'importe quel programme ?

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

1. Python : un langage de haut-niveau

2. Constructions élémentaires

Python : un langage de haut-niveau

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Un langage :

- ▶ développé par *Guido van Rossum* fin 1989,

Un langage :

- ▶ développé par *Guido van Rossum* fin 1989,
- ▶ interprété, c'est à dire que le programme est lu ligne après ligne par un *interpréteur*.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

À retenir

- ▶ Le *C* est un langage *compilé* : le **compilateur** construit un programme exécutable autonome à partir du code source.
- ▶ *Python* est un langage *interprété* : le code source est lu puis exécuté par l'**interpréteur**.

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Imaginons un programme qui demande à l'utilisateur de deviner la valeur du dé stockée en mémoire puis affiche le nombre d'essais nécessaires pour trouver la réponse.

Activité 1 : Proposer un algorithme qui réalise le jeu précédent.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

- ▶ Stocker la valeur du dé dans une **variable**.
- ▶ Demander à l'utilisateur d'**entrer** une valeur.
- ▶ **Répéter** :
 - ▶ **Comparer** la valeur de l'utilisateur à celle du dé.
 - ▶ Incrémenter le nombre d'essais.
- ▶ **Afficher (sortir)** le nombre d'essais dans la console.

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Activité 2 :

1. Dans le dossier *Maths* ou *NSI* du bureau, ouvrir une console Python : *Python 3.x.x* (figure 1).

```
Python 3.7.6 (default, Jan 8 2020, 19:59:22)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

FIGURE 1 – Console Python

2. Écrire les codes suivants puis valider. Observer les résultats obtenus.

1 3+4

1 3<4

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Il peut rapidement être fastidieux d'écrire un programme dans la console Python. Un **Environnement de Développement Intégré** permettra d'écrire plusieurs lignes de code puis se chargera d'envoyer toutes ces lignes à l'interpréteur. De plus, il sera possible d'enregistrer le programme dans un fichier.

Activité 3 :

1. Dans l'espace personnel de l'ordinateur, créer un dossier NSI
2. Ouvrir un EDI au choix : Spyder, Pyzo, EduPython.
3. Écrire le programme suivant dans la partie gauche de l'EDI.

```
1 print(3+4)
2 print(3<4)
```

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Activité 3 :

4. Enregistrer le programme dans le dossier *NSI* sous le nom `premier.py`
5. Exécuter le programme en cliquant sur la flèche verte (figure 2) ou en appuyant sur la touche *F5*. Le code est exécuté dans la console en bas à droite.



FIGURE 2 – Exécuter le programme

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés
Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

En première approche, nous pouvons considérer une **variable** comme une boîte qui contient une information. Cette information peut être de plusieurs natures (nombre, texte, tableau...)

Activité 4 :

1. Dans le dossier *NSI*, créer un fichier `jeu_de.py`
2. Écrire le code suivant :

```
1 de1 = 3
```

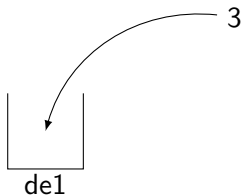


FIGURE 3 – Affectation

Remarque

Même si cela est possible en Python nous n'utiliserons pas de caractère accentué dans les noms de variables. De plus nous pourrons utiliser le signe `_` pour *coller* plusieurs mots dans un nom.

```
1 le_de = 3
```

Remarque

Le signe `=` n'est pas à comprendre au sens mathématique. C'est un signe d'affectation. Ainsi l'instruction

```
1 3 = a
```

est juste mathématiquement mais ne signifie rien en Python.

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

L'instruction

```
1 input()
```

attend une valeur de l'utilisateur. Cependant, la valeur est inaccessible. Il faut donc la stocker dans une variable :

```
1 proposition = input("Choisir une valeur du dé: ")
```


L'instruction

```
1 print("mon texte")
```

affiche *mon texte* à l'écran. Il est possible d'afficher le contenu d'une variable :

```
1 print(proposition)
```

Il ne faut alors pas mettre de guillemets.

Remarque

La valeur récupérée par `input` est une chaîne de caractère. Il est nécessaire d'insérer l'appel `input()` dans une fonction `int()` pour convertir la réponse en un entier :

```
1 proposition = int(input("Choisir: "))
```

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Pour vérifier la valeur du dé proposée par l'utilisateur, il faut la comparer avec celle du programme.

À retenir

L'instruction

```
1 a == b
```

renvoie **True** si les variables **a** et **b** sont égales, **False** sinon. Il faut noter l'emploi du **==** pour ne pas confondre avec le signe d'affectation.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

À retenir

Le mot-clé `if` évalue la comparaison qui le suit et agit en conséquence.

```
1  if a == b:
2      print("a et b sont égaux.")
3  else:
4      print("a et b sont différents.")
```

Code 2 – Comparaison (le `else` est facultatif).

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Activité 5 : Compléter le programme `jeu_de` pour que le message **gagné** s'affiche si la proposition de l'utilisateur est correcte, **perdu** sinon.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

```
1 if de1 == proposition:
2     print("Gagné")
3 else:
4     print("Perdu")
```

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

Boucle non bornée

Boucle bornée

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

À retenir

Une boucle non bornée répète une instruction tant que la condition est vérifiée.

Activité 6 :

1. Dans un nouveau fichier tester le programme ci-après :

```
1  compteur = 10
2  while compteur > 0:
3      print(compteur)
4      compteur = compteur - 1
5  print("Boum")
```

2. En anglais, que signifie **while** ?
3. À quelle ligne compare-t-on le compteur avec la valeur limite ?
4. Quel est le rôle de la ligne 4 ? Que se passera-t-il si cette ligne est retirée ?

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

- **Tant que** le contenu de la variable **compteur** est un entier supérieur à 10, les lignes 3 et 4 sont répétées.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

- ▶ **Tant que** le contenu de la variable **compteur** est un entier supérieur à 10, les lignes 3 et 4 sont répétées.
- ▶ La ligne 5 n'est pas dans la boucle : elle n'est pas *indentée*.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

- ▶ **Tant que** le contenu de la variable **compteur** est un entier supérieur à 10, les lignes 3 et 4 sont répétées.
- ▶ La ligne 5 n'est pas dans la boucle : elle n'est pas *indentée*.
- ▶ Si la ligne 4 est retirée, la boucle tourne indéfiniment.

Activité 7 :

1. Modifier le programme `jeu_de` pour qu'il interroge l'utilisateur tant que ce dernier ne donne pas la bonne réponse.
2. Ajouter un `compteur` qui compte le nombre de tentatives.

```
1  de1 = 3
2  proposition = 0
3  while de1 != proposition:
4      proposition = int(input("Choisir: "))
```

```
1  de1 = 3
2  proposition = 0
3  compteur = 0
4  while de1 != proposition:
5      compteur = compteur + 1
6      proposition = int(input("Choisir: "))
7  print(compteur)
```


Activité 8 :

1. Tester le programme ci-après :

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

2. Lire la documentation de la fonction *range* :

<https://tinyurl.com/rangepython>

3. Adapter le code précédent pour afficher :

- ▶ 6
- ▶ 7
- ▶ 8

4. Adapter le code précédent pour afficher :

- ▶ 0
- ▶ 3
- ▶ 6
- ▶ 9

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

```
1 for compteur in range(6,9):  
2     print(compteur)
```

Code 3 – Borne de départ

```
1 for compteur in range(0,10,3):  
2     print(compteur)
```

Code 4 – Pas

1. Python : un langage de haut-niveau

2. Constructions élémentaires

2.1 Jeu de dés

2.2 Interpréteur

2.3 Variables

2.4 Entrée/Sortie

2.5 Comparaison

2.6 Répétition

2.7 Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque

Le jeu de dés semblerait être plus intéressant si la réponse n'était pas *codée en dur* dans le programme.

Python fournit des **bibliothèques** spécialisées. La bibliothèque `random` permet de générer des nombres aléatoires.

```
1 from random import randint
2
3 de1 = randint(1,6)
```

Code 5 – Générer un entier aléatoire

Activité 9 : Modifier le programme `jeu_de` en utilisant le code 5.

Python : un
langage de
haut-niveau

Constructions
élémentaires

Jeu de dés

Interpréteur

Variables

Entrée/Sortie

Comparaison

Répétition

Boucle non bornée

Boucle bornée

Bibliothèque