
Décode
et Code
Bastien Gorissen & Thomas Stassin

Level 3-1

“BOSS BATTLE II : Electric Boogaloo”

A vous de jouer!

Nous allons réaliser un petit jeu de “pendu”.

Pour cela, il va falloir:

- Une liste de mots à trouver
- Une liste des lettres proposées par l’utilisateur
- Imprimer le mot à trouver avec uniquement les lettres devinées, et des “-” pour remplacer les autres.
- Détecter quand le joueur a trouvé toutes les lettres.

Étapes pour le pendu

- Créer une liste de mots à deviner
- Choisir un mot au hasard dans la liste
- Créer une variable string contenant autant de "-" que de lettres dans le mot
- Créer une liste vide pour stocker les propositions du joueur
- Tant que la variable est différente du mot:
  - Demander une proposition au joueur
  - L'ajouter dans une liste des propositions
  - Recréer la string avec un "-" pour chaque lettre inconnue, et les bonnes lettres si elles ont été proposées par l'utilisateur (partie compliquée :D)

Level 3-2

“Ray - Vision”

Exercice 1 (courage il y en a 22 :D)

Affectez la valeur 3 à un variable nommée **a**, ensuite affichez-la.

Exercice 2

A la suite du script précédent,

Affectez 3 à une variable nommée **b**

Et affichez la somme de **a** et **b**

Exercice 3

Dans un autre script:

Affectez la *chaîne de caractère* “Brutor” à la variable **name**, ensuite affichez cette variable, suivie de la *chaîne de caractères* “ II le retour”.

Exercice 4

A la suite du script précédent:

Affectez la *valeur* 10 à la variable **number**, ensuite affichez cette variable suivie d'un espace et encore suivie de la variable **name**.

Attention la variable **number** est un entier il faut donc la convertir en chaîne de caractères.

Exercice 5

Dans un nouveau script:

Stockez dans la variable **keyboard** le résultat d'une saisie au clavier et affichez-la.

Exercice 6

Dans le même script:

A la place d’afficher simplement la variable **keyboard**, affichez son contenu précédé de la chaîne de caractères “Saisie clavier: ”.

Exercice 7

Dans le même script:

A la place de l’affichage précédent, récupérez encore une fois la saisie au clavier mais cette fois dans la variable **other_keyboard.**

Affichez les deux variables séparées par un espace.

Exercice 8

Dans un autre script:

Récupérez au clavier un nombre et stockez-le dans la variable **number**.

Ensuite, affichez cette variable.

Rappelez-vous, la saisie clavier revient sous forme de chaîne de caractères.

Exercice 9

Dans le même script:

Si le nombre qui se trouve dans la variable **number** est plus grand que 10, affichez la chaîne de caractères “Ce nombre est plus grand que 10”.

Exercice 10

Dans le même script:

A la suite de la condition, si celle-ci n’est pas remplie, affichez le message suivant: “Le nombre est plus petit ou égal à 10.”

Exercice 11 (Vous êtes à la moitié!!)

Dans un autre script:

Récupérez un nombre au clavier et stockez-le dans une variable (choisissez le nom de la variable).

Si le nombre récupéré est entre 3 et 4 affichez le message “Boum” sinon affichez le message “Rien ne se passe”.

Exercice 12

Dans le même script:

Remplacez la saisie clavier par un nombre aléatoire entre 1 et 6.

Rappelez-vous qu'il faut importer *randint* à l'aide de la ligne de code suivante: *from random import randint*.

Cette ligne doit se trouver en première dans votre script.

La fonction *randint* a besoin qu'on lui passe deux nombres en argument, le minimum et le maximum.

Exercice 13

Dans un autre script:

Stockez dans une variable un nombre entre 0 et 10.

Ensuite affichez-le.

Si ce nombre est supérieur à 9, affichez le message “Score maximum”

Si le nombre est supérieur à 5 affichez le message “Bon score”

Dans les autres cas, affichez le message “score correct”

Attention, n’affichez qu’un seul message.

Exercice 14

Dans un autre script:

Imaginons que nous voulions stocker les points de vie d’un monstre dans une variable. De base le monstre a 30 points de vie.

Ensuite simulons une attaque en stockant la quantité de dégâts dans une autre variable. Les dégâts peuvent être de 5 à 10.

Ensuite soustrayez les dégâts aux points de vie du monstre.

Puis affichez les points de vie restants du monstre après l’attaque.

Exercice 15

Dans le même script:

Faites en sorte que les dégâts infligés aux points de vie du monstre (et donc la soustraction) soient appliqués 3 fois.

A chaque fois, affichez le nombre de points de vie avec une phrase du type: “Il reste x points de vie au monstre” (en remplaçant x par le nombre de points de vie du monstre).

Exercice 16

Dans le même script:

Au lieu de répéter l’opération 3 fois, répétez l’opération tant que le monstre est en vie, c’est-à-dire tant que ses points de vie sont supérieurs à 0.

Une fois que le monstre est mort, indiquez-le par un message.

Exercice 17

Dans un autre script:

Stockez dans une variable une liste de nombres allant de 0 à 5.

Affichez chaque élément de cette liste un à un.

Exercice 18

Dans un autre script:

Générez un nombre aléatoire entre 0 et 10 Bouclez tant que le nombre n’est pas égal à 0. A chaque itération stockez ce nombre dans une liste.

Ensuite affichez les nombres de la liste un à un.

Exercice 19

Dans le même script:

Changez la condition d’arrêt de la boucle. Au lieu de s’arrêter lorsque le nombre vaut 0, la boucle s’arrête quand le programme a généré deux fois le même nombre de suite.

Exercice 20

Dans un script différent:

Récupérez des mots saisis au clavier tant que le mot “end” n’a pas été saisi.

Ensuite affichez les mots dans l’ordre inverse de leur entrée.

Attention le mot “end” ne doit pas être récupéré (ni affiché).

Exercice 21

Dans un autre script:

Récupérez des entiers saisis au clavier. Faites le tant qu’un 0 n’est pas entré.

Après, affichez le plus petit et le plus grand nombre entrés de cette manière.

Exercice 22 (The last One)

Dans un autre script:

Saisissez des nombres au clavier.

Si le nombre entré est supérieur à 10 ou inférieur à 1 affichez un message qui indique que le nombre n’est pas accepté, sinon stockez le nombre.

Une fois que vous avez saisi 5 nombres affichez-les dans l’ordre de saisie.

LEVEL 3-3

“Le dictionnaire de A à Z”

Dict...

Un dictionnaire est un type de variable connu en Python sous le nom de **dict**.

Ce type de variable ressemble à une liste car il sert aussi à stocker plusieurs valeurs dans une même variable.

La différence avec sa cousine la liste c’est que le dictionnaire n’utilise pas des index qui se basent sur la position dans la liste, mais des clés.

Le dictionnaire clé en main

Chaque valeur du dictionnaire est liée à une clé qui est unique et chacune est liée à une valeur

Si, par exemple, je veux stocker dans un dictionnaire les données d'une recette de potion avec en clés les ingrédients et valeurs la quantité dont j'en ai besoin, j'écrirai:

recipe = {'toad bubble': 1, 'rat tongue': 2, 'spider jaw': 1}

Le dictionnaire clé en main

Donc oui, pour écrire un dictionnaire dans le code, on écrit les données entre accolades de la manière suivante: “*clé: valeur*”, chacune des données étant séparée par une virgule.

Dans l’exemple de la potion:

On retrouve bien les **clés** et les **valeurs**

```
{‘toad bubble’: 1, ‘rat tongue’: 2, ‘spider jaw’: 1}
```

NB: un dictionnaire vide s’écrit comme ceci: **{}**

Le dictionnaire clé en main

Les clés et les valeurs peuvent être de tout type:

Par exemple on pourrait décider d'utiliser des entiers comme clés qui représentent un numéro d’identification et comme valeur le nom de la personne derrière ce numéro d'identification.

De la même façon, on peut mettre comme valeur n’importe quel type de données, par exemple un **str**, un **int**, une **list** ou même… un autre **dict**

Lire un dictionnaire

Si on veut lire une donnée particulière du dictionnaire on, procède comme pour la **list** à la différence près que l'on met entre les crochets la clé et non l'index.

Pour l'exemple de la potion, si je veux afficher la quantité de langues de rat j'écrirais:

```
print(recipe['rat tongue'])
```

Ecriture d'un dictionnaire

De la même manière, si je veux changer la quantité de langues de rat j'écrirais:

recipe['rat tongue'] = 5

Attention, si vous tentez d'écrire une donnée pour une clé qui n'existe pas, elle sera créée:

recipe['dragon bone'] = 2

Cette ligne crée la clé 'dragon bone' et y associe la valeur 2.

Le dictionnaire ressemblera ensuite à:

{'toad bubble': 1, 'rat tongue': 5, 'spider jaw': 1, 'dragon bone': 2}

Clé inconnue

Par contre, si vous tentez d'accéder à une clé qui n'existe pas sans passer par l'affectation (par exemple en tentant de l'imprimer), Python renverra une erreur.

Et maintenant à vous de jouer...

Le jeu du dictionnaire

Imaginons que l'on veuille coder les prémisses du nouveau Dragon Age.

Dans un script vierge :

Entrez dans un dictionnaire par saisie clavier, le nom du héros sous la clé 'name' et sa puissance sous la clé 'power'.
Ensuite, affichez ces données à l'écran.

Level 3-4

“Boucle et dictionnaire”

On boucle tout :D

A l'instar de sa cousine la **list**, on peut boucler sur un dictionnaire à l'aide d'une boucle **for**.

Quelques différences toutefois.

On boucle tout :D

Si j’écris:

```
inventory = {'potion ': 1, 'scroll ': 2, 'holy sword ': 0}

for key in inventory :

    print (key)
```

A l’écran va apparaître:

```
potion

holy sword

scroll
```

On boucle tout :D

Lorsqu'on boucle sur un dictionnaire, ce qui va dans la variable d'itération (dans ce cas-ci *key*), ce sont les clés du dictionnaire.

Notez que celles-ci n'arrivent pas dans l'ordre dans lequel on les a inscrites.

De fait, le dictionnaire ne mémorise pas l'ordre d'entrée des données.

Et si je veux les valeurs alors?

On boucle tout :D

Si l’on veut boucler sur les valeurs, on passera par la *méthode* values() propre au dictionnaire.

```
inventory = {'potion ': 1, 'scroll ': 2, 'holy sword ': 0}

for value in inventory.values():

    print( value )
```

Dans ce cas là j’obtiens les valeurs à la place des clés.

Et si je veux les deux?

On boucle tout :D

Si l'on veut boucler sur les clé et les valeurs, on passera par la *méthode* items() elle aussi propre au dictionnaire.

```
inventory = {'potion ': 1, 'scroll ': 2, 'holy sword ': 0}

for key, value in inventory.items():

    print(key + ' :' + str(value))
```

Notez que dans ce cas là j'ai besoin de deux variables d'itération une pour la clé une pour la valeur

And now… Exercice!!

Dragon Age Dictionary

Toujours dans l'idée de faire le nouveau Dragon Age.

Stockez les caractéristiques du héro dans un dictionnaire :

La puissance vaut 10, la magie vaut 5 et la défense vaut 3.

Ensuite demandez à l'utilisateur le nom du héros et stockez-le dans le dictionnaire.

Enfin bouclez sur ce même dictionnaire et affichez toutes les caractéristiques et leurs valeurs (donc clé et valeur).