

Séance 1 : VARIABLES, FONCTIONS ET CONDITIONS

L1 – Université Côte d'Azur

Pour ce TP, nous vous demandons d'utiliser l'éditeur Thonny. On vous conseille fortement de l'installer sur votre ordinateur personnel pour travailler chez vous. Vous pouvez le télécharger à l'adresse : <https://thonny.org/>. Si vous tenez vraiment à utiliser un autre éditeur (GNU/Emacs, notepad++, Idle, etc.), ce sera à vous de vous assurer que tout marche correctement ! Nous vous conseillons cependant fortement de rester avec Thonny.

Création du répertoire de travail

Avant de commencer ce TP : sur votre ordinateur, créez un répertoire BI1 dans lequel vous rangerez tous vos fichiers de l'UE. Dans votre répertoire BI1 créez un répertoire TP1. Durant cette séance, tous vos fichiers devront être sauvegardés dans ce répertoire TP1.

Les prochaines semaines, vous devrez créer les répertoires TP2, puis TP3. Ainsi il sera facile de vous retrouver dans tous les fichiers.

Exercice 1 – Premiers pas dans la console

Cette partie doit se faire entièrement dans la console (avec les lignes commençant par >>>). Toutes les réponses tiennent en une ligne (vous ne pouvez pas définir de nouvelles fonctions, ni utiliser de `if`).

1. Définissez deux variables : `p` ayant pour valeur 5 et `q` ayant pour valeur 3p.
2. Avec une seule instruction `print(...)` et en utilisant les variables `p` et `q`, faites afficher la phrase suivante : `p vaut 5 et q vaut 15 ; leur somme fait 20`.

N.B. Votre code doit fonctionner même si `p` et `q` ont des valeurs différentes. Ainsi, la solution suivante n'est *pas* celle que l'on attend : `print('p vaut 5 et q vaut 15 ; leur somme fait 20')`.

3. Demander en python « `q est-il un multiple de p ?` ». Indication : utilisez le reste de la division euclidienne. Votre calcul doit renvoyer `True` ou `False`.
4. Sans consulter vos notes de cours, écrivez dans la console les lignes de code permettant d'échanger les valeurs de `p` et `q`, en utilisant une variable temporaire `tmp`. Vérifiez ensuite dans la fenêtre Variables¹ de Thonny que les valeurs de `p` et `q` ont bien été permutées.
5. Calculez la moyenne des entiers de 1 à 10. Le résultat est un nombre flottant (à virgule).
6. Calculez la moyenne des entiers de 1 à 11. Ne retapez pas la formule complète, mais modifiez la précédente en tapant sur la touche du haut `↑` qui va chercher les lignes précédemment entrées (l'*historique*). On peut utiliser `↓` en sens inverse.
7. Comment feriez-vous pour savoir si la fraction $\frac{51}{85}$ est irréductible ? En d'autres termes, peut-on la simplifier ? Par combien ?
8. Demandez en une ligne si 5^4 est plus grand que 4^5 . Le résultat devra être un booléen.
9. Tapez dans la console : `help(max)` pour demander une petite documentation de la fonction `max`.

1. Si vous ne voyez pas de fenêtre « variable », allez dans le menu « afficher » de Thonny et puis cliquez sur l'élément « Variables »

Exercice 2 — Utilisation de Thonny en mode debug

1. Relisez le cours 1 partie II les deux pages « Thonny : écrire des scripts » et « Thonny : raccourcis clavier ». Vous le trouverez sur ma page professionnelle :

<https://upinfo.univ-cotedazur.fr/~obaldellon/python>

2. Créez un fichier `exercice2.py` dans votre dossier TP1 que vous avez créé au début du TP. Ce fichier devra contenir le code suivant.

```
1 r = 0
2 m = int(input("Entrer un nombre entre 0 et 999 :"))
3
4 c = m%10
5 r = r*10 + c
6 m = m//10
7 print(c)
8
9 c = m%10
10 r = r*10 + c
11 m = m//10
12 print(c)
13
14 c = m%10
15 r = r*10 + c
16 m = m//10
17
18 print(c)
19 print(r)
```

3. Exécutez-le en appuyant sur la touche `[F5]`. Que fait-il ?
4. Exécutez-le pas à pas en appuyant sur les touches `[⇧]+[F5]` puis en appuyant plusieurs fois sur la touche `[F7]`. Observez.
5. Exécutez-le pas à pas en appuyant sur les touches `[Ctrl]+[F5]` (pour passer en mode debug) puis en appuyant plusieurs fois sur la touche `[F7]`. Observez. Quelle est la différence ?

Exercice 3 — Générer une facture

Un client veut acheter n produits de prix p . Le prix final est donc $n \times p$. Il faut cependant ajouter le prix de la livraison et une éventuelle réduction.

1. À partir de $n = 10$, on applique une réduction de 10% et à partir de 50, la réduction devient 20%. Écrire la fonction `réduction(n)` qui renvoie selon la valeur de n une des valeurs parmi : 0, 10 ou 20. **Écrire au moins cinq tests.**

```
1 def réduction(n):
2     if n>=50:
3         return 20
4     elif n>=10:
5         return 10
6     else:
7         return 0
8
9 assert réduction(100) == 20
10 assert réduction(50) == 20
11 assert réduction(49) == 10
12 assert réduction(10) == 10
13 assert réduction(9) == 0
```

2. Le prix de la livraison correspond à 10% du prix initial (avant la réduction). Cependant, le prix de la livraison obtenue ne peut pas être inférieur à 5€ ou supérieur à 100€. Par exemple si le calcul des 10% donne moins que 5€

on fixera le prix de la livraison à 5€. Écrire la fonction `prix_livraison(prix)` qui prend en argument le prix initial et renvoie le prix de livraison. **Écrire au moins cinq tests.**

```

1 def prix_livraison(prix):
2     prix_l = prix*0.1 # 10% de prix
3     if prix_l > 100:
4         return 100
5     elif prix_l < 5:
6         return 5
7     else:
8         return prix_l
9
10 assert prix_livraison(200) == 100
11 assert prix_livraison(1000) == 100
12 assert prix_livraison(1001) == 100
13 assert prix_livraison(50) == 5
14 assert prix_livraison(49.5) == 5

```

3. Écrivez un programme `facture(n,p)` qui calcule le prix final en affichant les détails du calcul comme ci-dessous.

```

1 >>> facture(101,3)
2 Prix : 303 €
3 Réduction : - 20 %
4 Coût livraison : 30.3 €
5 -----
6 Prix total (dont livraison) : 272.7 €

```

```

1 def facture(n,p):
2     prix = n*p
3     r = reduction(n)
4     l = prix_livraison(prix)
5     prix_final = prix * (1-r/100)+l
6     print("Prix :",prix,"€")
7     print("Réduction : -",r,"%")
8     print("Coût livraison :",l,"€ ")
9     print("-----")
10    print("Prix total (dont livraison) :",prix_final,"€")

```

Exercice 4 — Pierre-Feuille-Ciseaux

Le jeu pierre-feuille-ciseaux (aussi appelé parfois *mourre* en Italie ou en Chine) est un jeu de mains où deux joueurs s'affrontent en choisissant simultanément chacun un objet parmi *pierre*, *feuille* ou *ciseaux*. Si les deux joueurs choisissent le même objet, il y a égalité, sinon la feuille l'emporte sur la pierre, qui l'emporte sur les ciseaux, qui l'emportent sur la feuille. Vous allez maintenant programmer un jeu de mourre pour affronter votre ordinateur !

1. Créez un fichier `exercice4.py` dans votre dossier TP1.
2. Écrivez un script qui affiche la phrase « Quel est ton choix, humain ? » puis qui demande à l'utilisateur de saisir un texte, et enfin affiche la phrase « Humain, tu as choisi XXXX », où XXXX est remplacé par le texte saisi par l'utilisateur juste avant. Par exemple :

```

1 Quel est ton choix, humain ? pierre
2 Humain, tu as choisi pierre

```

Vous utiliserez la fonction `input(msg)` qui affiche le message contenu dans le paramètre `msg` et renvoie ce que l'utilisateur a saisi au clavier à la suite de ce message.

```

1 choix_humain = input('quel est ton choix, humain ? ')
2 print('Humain, tu as choisi', choix_humain)

```

3. Modifiez votre programme pour qu'il affiche la phrase « Ok » si l'utilisateur a bien saisi l'un des trois mots *pierre*, *feuille*, ou *ciseaux*, et sinon affiche la phrase « Choix incorrect, je vais choisir pour toi... ». Pour l'instant on affiche juste le message, le choix sera fait dans les questions suivantes. Par exemple, on aura :

```
1 Quel est ton choix, humain ? pierre
2 Ok
```

ou encore

```
1 Quel est ton choix, humain ? puits
2 Choix incorrect, je vais choisir pour toi...
```

```
1 if choix_humain=='pierre' or choix_humain=='feuille' or choix_humain=='ciseaux':
2     print('ok')
3 else :
4     print('choix incorrect, je choisis pour toi...')
```

4. Revenez au début du programme et définissez une fonction `choix_ordi()` qui renvoie un mot au hasard parmi les trois mots possibles (vous aurez besoin de la fonction `randint` du module `random` vue en cours sur le transparent du cours intitulé : « Aléatoire : générer des entiers au hasard »). Complétez votre programme pour que le message d'erreur précédent indique le choix qu'a fait l'ordinateur. On aura désormais :

```
1 Quel est ton choix, humain ? puits
2 Choix incorrect, je choisis pour toi : feuille
```

```
1 from random import randint
2
3 def choix_ordi():
4     choix = randint(0,2)
5     if choix == 0 :
6         return 'pierre'
7     elif choix == 1 :
8         return 'feuille'
9     else :
10        return 'ciseaux'
```

```
1 if choix_humain=='pierre' or choix_humain=='feuille' or choix_humain=='ciseaux':
2     print('ok')
3 else :
4     choix_humain = choix_ordi()
5     print('Choix incorrect, je choisis pour toi : ', choix_humain)
```

5. Complétez votre programme pour faire une partie complète : l'ordinateur fait un choix pour lui au début (mais le garde secret), puis l'utilisateur fait un choix, puis le programme affiche les deux choix et le gagnant éventuel. Par exemple :

```
1 Quel est ton choix, humain ? ciseaux
2 ok
3 Humain : ciseaux
4 Ordinateur : feuille
5 Vainqueur : humain
```

Avant de déterminer le gagnant, on affiche les différents choix des deux joueurs.

```
1 choix_ordinateur = choix_ordi()
2 print('Humain : ',choix_humain)
3 print('Ordinateur : ',choix_ordinateur)
```

Solution 1 : on énumère tous les cas. C'est long et pénible (9 cas), à éviter

```
1 if choix_humain == 'pierre' and choix_ordinateur == 'pierre' :
2     print('Vainqueur : égalite')
3 if choix_humain == 'pierre' and choix_ordinateur == 'feuille' :
4     print('Vainqueur : ordi')
5 if choix_humain == 'pierre' and choix_ordinateur == 'ciseaux' :
6     print('Vainqueur : humain')
7 # ...
8 # Et on continue comme ça pour les 6 autres cas. C'est fastidieux
9 # En info, il faut être paresseux et chercher à faire plus simple.
```

Solution 2 : on énumère tous les cas, mais cette fois-ci en regroupant les tests avec deux niveaux de `if`. C'est déjà beaucoup plus lisible, mais ça reste laborieux.

```
1 if choix_humain == 'feuille' :
2     if choix_ordinateur == 'pierre' :
3         print('Vainqueur : humain')
4     if choix_ordinateur == 'feuille' :
5         print('Vainqueur : égalite')
6     if choix_ordinateur == 'ciseaux' :
7         print('Vainqueur : ordi')
8
9 if choix_humain == 'ciseaux' :
10     ...
```

Solution 3 : on ne fait plus que trois cas : l'égalité, la victoire de l'humain et sa défaite. Remarquez que l'on peut créer des variables booléennes (c'est à dire `True` ou `False`) `cas_gagnant_1`, `cas_gagnant_2` ou `cas_gagnant_3` pour rendre le test plus lisible.

```
1 # Quels sont les trois cas qui font gagner l'être humain ?
2 cas_gagnant_1 = (choix_humain == 'pierre' and choix_ordinateur == 'ciseaux')
3 cas_gagnant_2 = (choix_humain == 'feuille' and choix_ordinateur == 'pierre')
4 cas_gagnant_3 = (choix_humain == 'ciseaux' and choix_ordinateur == 'feuille')
5
6 if choix_humain == choix_ordinateur :
7     print('Vainqueur : égalite')
8 elif cas_gagnant_1 or cas_gagnant_2 or cas_gagnant_3:
9     print('Vainqueur : humain')
10 else :
11     print('Vainqueur : ordinateur')
```

Version Complète

```
1 from random import randint
2
3 def choix_ordi():
4     choix = randint(0,2)
5     if choix == 0 :
6         return 'pierre'
7     elif choix == 1 :
8         return 'feuille'
9     else :
10        return 'ciseaux'
11
12 choix_humain = input('quel est ton choix, humain ? ')
13 print('Humain, tu as choisi', choix_humain)
14
15 if choix_humain=='pierre' or choix_humain=='feuille' or choix_humain=='ciseaux':
16     print('ok')
17 else :
18     choix_humain = choix_ordi()
19     print('Choix incorrect, je choisis pour toi : ', choix_humain)
20
21 choix_ordinateur = choix_ordi()
22 print('Humain : ',choix_humain)
23 print('Ordinateur : ',choix_ordinateur)
24
25 # Quels sont les trois cas qui font gagner l'être humain ?
26 cas_gagnant_1 = (choix_humain == 'pierre' and choix_ordinateur == 'ciseaux')
27 cas_gagnant_2 = (choix_humain == 'feuille' and choix_ordinateur == 'pierre')
28 cas_gagnant_3 = (choix_humain == 'ciseaux' and choix_ordinateur == 'feuille')
29
30 if choix_humain == choix_ordinateur :
31     print('Vainqueur : égalite')
32 elif cas_gagnant_1 or cas_gagnant_2 or cas_gagnant_3:
33     print('Vainqueur : humain')
34 else :
35     print('Vainqueur : ordinateur')
```

Exercice 5 — Le juste prix

Le juste prix consiste à deviner le prix d'un objet (un entier inférieur ou égal à 100 euros) en faisant des propositions de prix, l'autre joueur indiquant si on est au-dessus ou en dessous du prix réel. On commencera par créer un fichier `exercice5.py`.

1. En vous inspirant de l'exercice précédent, programmez le jeu du juste prix où c'est l'humain qui doit deviner le prix. Il a droit à 7 propositions. Indication : `int('42') == 42`.

On définit une fonction auxiliaire `juste_prix_essais(n)` où n correspond aux nombres de tentatives restantes. Si $n = 0$, il n'y a plus de tentatives. Au départ $n = 7$. Cette fonction auxiliaire renvoie un booléen qui dit si on a trouvé (`True`) ou pas (`False`).

```
1 def essai(prix):
2     pronostic = int(input('Quel est votre pronostic ?'))
3     if pronostic == prix:
4         print('Trop bien !')
5         return True
6     elif pronostic > prix:
7         print('Trop haut !')
8         return False
9     else:
10        print('Trop bas !')
11        return False
12
13
14 def juste_prix_essais(n,prix):
15     if n == 0:
16         return False
17     else:
18         tentative = essai(prix)
19         if tentative:
20             return True
21         else:
22             return juste_prix_essais(n-1,prix)
23
24 # On peut écrire juste_prix_essais de manière plus dense.
25 # À vous de choisir la méthode qui vous semble la plus simple à comprendre.
26 def juste_prix_essais(n,prix):
27     if n == 0:
28         return False
29     else:
30         return essai(prix) or juste_prix_essais(n-1,prix)
31
32
33
34 def juste_prix():
35     prix = randint(0,100)
36     trouvé = juste_prix_essais(7,prix)
37     if trouvé:
38         print('Gagné !')
39     else:
40         print('Essais épuisés, perdu, la valeur était', prix,trouvé)
```

2. Quelle stratégie adoptez-vous pour gagner à coup sûr ?

La stratégie à adopter est une dichotomie, on connaît les valeurs les plus grandes et petites possibles, on teste au milieu. 7 essais suffisent car la longueur de l'intervalle est divisée par 2 à chaque étape. En partant d'un intervalle de longueur

100, on obtient au plus un intervalle réduit à un élément en 7 étapes.

3. Programmez le jeu où c'est l'ordinateur qui fait des propositions et l'humain qui après avoir choisi un nombre entre 0 et 100 répond « + » ou « - » selon la valeur proposée par l'ordinateur. Si l'ordinateur a trouvé la solution, vous répondrez « = ». L'ordinateur a droit à 7 propositions et devra gagner à coup sûr ou afficher un message spécial si l'humain a triché dans ses réponses.

```
1 def essai_ordinateur(valeur_min, valeur_max):
2     if valeur_min > valeur_max:
3         return False
4     essai = (valeur_min + valeur_max) // 2
5     print('Je devine', essai)
6     estimation = input()
7     if estimation == '=':
8         return True
9     elif estimation == '+':
10        return essai_ordinateur(essai + 1, valeur_max)
11    else: # estimation == '-'
12        return essai_ordinateur(valeur_min, essai - 1)
13
14
15 def juste_prix2():
16     trouvé = essai_ordinateur(0,100)
17     if not trouvé:
18         print('Tricheur !')
```