

**Programmation 2**

TP 2

*Lisez attentivement l'ensemble du document.**Nabil Mustafa***Consignes.**

(a) **L'objectif est d'apprendre la programmation en Python.**

**Il n'y a pas besoin de se précipiter !**

**Prenez votre temps et terminez correctement tous les TP précédents avant de commencer avec ce TP.**

(b) *Vous devriez lire attentivement les diapositives, pour réviser ce que nous avons fait en CM.*

(c) *Il est important de se concentrer et de lire le texte attentivement et lentement. Si quelque chose dans le texte n'est pas clair après l'avoir lu plusieurs fois, demandez au professeur.*

### Tâche 1: Lec2-conditionnelles.ipynb

- (a) Connectez-vous à LINUX avec votre compte utilisateur.
- (b) Téléchargez le Jupyter Notebook fichier [Lec2-conditionnelles.ipynb](#), disponible sur le site web du module.
- (c) Pour le lire, ouvrez un terminal dans le *même dossier* que le fichier [Lec2-conditionnelles.ipynb](#), et exécuter le command :

```
jupyter-notebook Lec2-conditionnelles.ipynb
```



- (d) Pour exécuter un code Python dans le Jupyter Notebook, cliquez sur le code et appuyez sur "SHIFT + ENTER".
- (e) **Lisez toutes les informations** dans cette Jupyter Notebook.
- (f) **Faites tous les exercices** dans cette Jupyter Notebook.

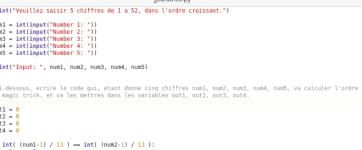
## Tâche 2: Exécuter Python

- (a) Connectez-vous à LINUX avec votre compte utilisateur.
  - (b) Téléchargez le fichier `5cardtrick.py`.
  - (c) Vous pouvez éditer ce fichier avec l'éditeur de votre choix. Par exemple, ouvrez un terminal dans le même dossier que le fichier `5cardtrick.py`, et exécuter le command:

kate 5cardtrick.py

- (d) Pour exécuter le code avec Python, ouvrez un terminal dans le même dossier que le fichier `5cardtrick.py`, et exécuter le command :

python3 5cardtrick.py



ScintillaPython - Kode

Eicher Edition Affichage Projets Signets Sessions Options Configuration Aide

Scintilla.py

```
print("Veuillez saisir 5 chiffres de 1 à 52, dans l'ordre croissant.")
```

```
num1 = int(input("Nombre 1 :"))
num2 = int(input("Nombre 2 :"))
num3 = int(input("Nombre 3 :"))
num4 = int(input("Nombre 4 :"))
num5 = int(input("Nombre 5 :"))

print("Input : ", num1, num2, num3, num4, num5)
```

```
#1: dessous, écrivez le code qui, ayant donné cinq chiffres num1, num2, num3, num4, va calculer l'ordre pour
#   la magie trix, et va les mettre dans les variables out1, out2, out3, out4.
```

```
[#1]
out1 = 0
out2 = 0
out3 = 0
out4 = 0

# If test (num1-1) / 13 == 0 then (num2-1) / 13 == 1:
if (num1-1) % 13 == 0 and (num2-1) % 13 == 1:
    a0 = num1
    a0 = num2
    a2 = num3
    a3 = num4
    a4 = num5

# If test (num1-1) / 13 == 1 then (num3-1) / 13 == 1:
if (num1-1) % 13 == 1 and (num3-1) % 13 == 1:
    a0 = num1
    a0 = num3
    a2 = num4
    a3 = num5
    a4 = num2

# If test (num1-1) / 13 == 2 then (num4-1) / 13 == 1:
if (num1-1) % 13 == 2 and (num4-1) % 13 == 1:
    a0 = num1
    a0 = num4
    a2 = num5
    a3 = num2
    a4 = num3

# If test (num1-1) / 13 == 3 then (num5-1) / 13 == 1:
if (num1-1) % 13 == 3 and (num5-1) % 13 == 1:
    a0 = num1
    a0 = num5
    a2 = num3
    a3 = num4
    a4 = num2

# If test (num2-1) / 13 == 0 then (num3-1) / 13 == 1:
if (num2-1) % 13 == 0 and (num3-1) % 13 == 1:
    a0 = num2
    a0 = num3
    a2 = num4
    a3 = num5
    a4 = num1

# If test (num2-1) / 13 == 1 then (num4-1) / 13 == 1:
if (num2-1) % 13 == 1 and (num4-1) % 13 == 1:
    a0 = num2
    a0 = num4
    a2 = num5
    a3 = num1
    a4 = num3

# If test (num2-1) / 13 == 2 then (num5-1) / 13 == 1:
if (num2-1) % 13 == 2 and (num5-1) % 13 == 1:
    a0 = num2
    a0 = num5
    a2 = num1
    a3 = num4
    a4 = num3

# If test (num3-1) / 13 == 0 then (num4-1) / 13 == 1:
if (num3-1) % 13 == 0 and (num4-1) % 13 == 1:
    a0 = num3
    a0 = num4
    a2 = num5
    a3 = num1
    a4 = num2

# If test (num3-1) / 13 == 1 then (num5-1) / 13 == 1:
if (num3-1) % 13 == 1 and (num5-1) % 13 == 1:
    a0 = num3
    a0 = num5
    a2 = num1
    a3 = num2
    a4 = num4

# If test (num3-1) / 13 == 2 then (num1-1) / 13 == 1:
if (num3-1) % 13 == 2 and (num1-1) % 13 == 1:
    a0 = num3
    a0 = num1
    a2 = num4
    a3 = num5
    a4 = num2

# If test (num4-1) / 13 == 0 then (num5-1) / 13 == 1:
if (num4-1) % 13 == 0 and (num5-1) % 13 == 1:
    a0 = num4
    a0 = num5
    a2 = num1
    a3 = num2
    a4 = num3

# If test (num4-1) / 13 == 1 then (num1-1) / 13 == 1:
if (num4-1) % 13 == 1 and (num1-1) % 13 == 1:
    a0 = num4
    a0 = num1
    a2 = num5
    a3 = num2
    a4 = num3

# If test (num4-1) / 13 == 2 then (num2-1) / 13 == 1:
if (num4-1) % 13 == 2 and (num2-1) % 13 == 1:
    a0 = num4
    a0 = num2
    a2 = num5
    a3 = num1
    a4 = num3

# If test (num5-1) / 13 == 0 then (num1-1) / 13 == 1:
if (num5-1) % 13 == 0 and (num1-1) % 13 == 1:
    a0 = num5
    a0 = num1
    a2 = num4
    a3 = num3
    a4 = num2

# If test (num5-1) / 13 == 1 then (num2-1) / 13 == 1:
if (num5-1) % 13 == 1 and (num2-1) % 13 == 1:
    a0 = num5
    a0 = num2
    a2 = num1
    a3 = num4
    a4 = num3

# If test (num5-1) / 13 == 2 then (num3-1) / 13 == 1:
if (num5-1) % 13 == 2 and (num3-1) % 13 == 1:
    a0 = num5
    a0 = num3
    a2 = num4
    a3 = num1
    a4 = num2

Ligne 13, colonne 1
```

INERTION F\_Fn Tabulations douces - 4 UTT-B Python

Veuillez saisir 5 chiffres de 1 à 52, dans l'ordre croissant.

Nombre : 1

## Rappel: ce que nous avons fait en CM.

Étant donné 5 nombres distincts—chacun entre 1 et 52—il y a un algorithme pour masquer l'un de ces 5 nombres et calculer un ordre pour les 4 nombres restants, de sorte qu'il soit possible de calculer le nombre qui est caché en examinant l'ordre des quatre nombres!

Pour illustrer l'algorithme, prenons les 5 nombres

2    15    30    38    50

L'algorithme est le suivant:

- a) Partitionnez *tous* les nombres entre 1 et 52 en quatre intervalles suivants:

[1 – 13]                  [14 – 26]                  [27 – 39]                  [40 – 52]

- b) Étant donné les 5 nombres, choisissez deux nombres qui appartiennent au même intervalle.

Dans notre exemple, les nombres 30 et 38 appartiennent à l'intervalle [27 – 39].

- c) L'un de ces deux nombres deviendra le premier nombre dans notre ordre, et nous cacherons le deuxième.

Dans notre exemple, 30 ou 38 sera le premier nombre, et nous cacherons le deuxième.

- d) Des deux nombres qui appartiennent au même intervalle, nous allons choisir le premier nombre pour notre ordre, tel que la **distance circulaire** de ce premier nombre au deuxième nombre est inférieur ou égal à 6.

Dans notre exemple :

la distance circulaire 30 → 38 est 8: 30 → 31 → 32 → 33 → 34 → 35 → 36 → 37 → 38.

la distance circulaire 38 → 30 est 5: 38 → 39 → 27 → 28 → 29 → 30.

Donc le premier nombre dans notre ordre va être 38 et on va cacher 30.

- e) Nous encodons cette distance en permutant les 3 nombres restants. Disons que les trois nombres sont  $a$ ,  $b$  et  $c$  en ordre croissant (donc  $a < b < c$ ). On va mettre  $a$ ,  $b$  et  $c$  dans un ordre comme suit. Je dispose de l'information suivante, une fois les trois autres nombres sont triés:

		2e	3e	4e
distance 1	⇒	$a$	$b$	$c$
distance 2	⇒	$a$	$c$	$b$
distance 3	⇒	$b$	$a$	$c$
distance 4	⇒	$b$	$c$	$a$
distance 5	⇒	$c$	$a$	$b$
distance 6	⇒	$c$	$b$	$a$

Dans notre exemple, les trois nombres restants sont 2 15 50. Donc  $a = 2$ ,  $b = 15$ ,  $c = 50$ . Il faut encoder la distance 5 avec l'ordre  $c \ a \ b$ , et donc l'ordre pour les trois nombres restants ( $a$ ,  $b$ ,  $c$ ) est : 50 2 15.

- f) Dans notre exemple, l'ordre finale est :

38    50    2    15.

et on va cacher le nombre 30.

## Tour de magie: exemples.

**Exemple 1:**

a) Les 5 nombres sont :

4    12    26    28    46

b) 4 et 12 appartiennent à la même catégorie

c) Dans l'ordre circulaire, 12 → 4 a distance 5 : 12 → 13 → 1 → 2 → 3 → 4.

d) On encode la distance 5 avec les 3 nombres restants,  $a = 26, b = 28, c = 46$  :

distance 5     $\Rightarrow$     46    26    28

Donc l'ordre est : 12    46    26    28. Avec cet ordre, il est possible de calculer le 5ème nombre.

**Exemple 2:**

a) Les 5 nombres sont :

1    16    33    36    52

b) 36 et 33 appartiennent à la même catégorie

c) Dans l'ordre circulaire, 33 → 36 a distance 3 : 33 → 34 → 35 → 36.

d) On encode la distance 3 avec les 3 nombres restants,  $a = 1, b = 16, c = 52$  :

distance 3     $\Rightarrow$     16    1    52

Donc l'ordre est : 33    16    1    52. Avec cet ordre, il est possible de calculer le 5ème nombre.

**Exemple 3:**

a) Les 5 nombres sont :

7    18    30    40    51

b) 40 et 51 appartiennent à la même catégorie

c) Dans l'ordre circulaire, 51 → 40 a distance 2 : 51 → 52 → 40.

d) On encode la distance 3 avec les 3 nombres restants,  $a = 7, b = 18, c = 30$  :

distance 2     $\Rightarrow$     7    30    18

Donc l'ordre est : 51    7    30    18. Avec cet ordre, il est possible de calculer le 5ème nombre.

### Tâche 3: Encodage

Dans le fichier `5cardtrick.py`, ajouter le code qui

1. prend les 5 chiffres `num[0]`, `num[1]`, `num[2]`, `num[3]`, `num[4]`,
2. calcule l'ordre pour le magic trick, et les met dans les variables `output[0]`, `output[1]`, `output[2]`, `output[3]`.

**Vous n'avez pas le droit d'utiliser de boucles.**

Voici l'exemple :

```
$ python3 5cardtrick_sol.py
Veuillez saisir 5 chiffres de 1 à 52, dans l'ordre croissant.
Number 1: 5
Number 2: 18
Number 3: 29
Number 4: 37
Number 5: 40
Input: 5 18 29 37 40
Les quatre premiers chiffres sont : 37 40 5 18
5ème chiffre: 29
```

### Indices.

Étape 1 : Identifier les deux nombres dans le même intervalle circulaire.

- Pour vérifier si `num[1]` et `num[2]` appartiennent au premier intervalle, on peut utiliser :  
`if (num[1] >= 1 and num[1] <= 13 and num[2] >= 1 and num[2] <= 13)`
- En utilisant une logique similaire, vous pouvez vérifier si `num[1]` et `num[2]` appartiennent au deuxième intervalle, ou au troisième intervalle, ou au quatrième intervalle.
- Vous pouvez vérifier n'importe quelle paire de nombres de cette façon.

Étape 2 : Calculer la distance minimale sur le cercle.

- Soient  $x$  et  $y$  les deux nombres trouvés à l'étape 1, avec  $x < y$ .
- **Distance directe** :  $d1 = y - x$ .    **Distance inverse** :  $d2 = \dots$  ?
- **Distance minimale** :  $dmin = \min(d1, d2)$ .

Étape 3 : Déterminer l'ordre d'émission.

- Si  $dmin = d1$ , alors on émet d'abord  $x$  (en position 1), puis on cache  $y$  en 5-ème.
- Si  $dmin = d2$ , alors on émet d'abord  $y$  (en position 1), puis on cache  $x$  en 5-ème.

Étape 4 : Permutation des trois nombres restants.

- Après avoir choisi quel nombre cacher, il reste trois nombres à transmettre.
- Maintenant, déterminez l'ordre de ces trois nombres aux positions 2, 3 et 4 en consultant la table d'encodage fournie précédemment, pour trouver la permutation correspondant à la distance  $dmin$ .

#### Tâche 4: Décodage

À la fin du même fichier `5cardtrick.py`, écrire le code qui prend les 4 premiers chiffres `output[0]`, `output[1]`, `output[2]`, `output[3]`, et calcule le 5-eme chiffre `output[4]`.