ACTIVITÉ 2: CONDITIONS ET BOUCLES EN PYTHON

Problématique: Comment programmer des instructions qui s'exécutent sous certaines conditions ou se répètent plusieurs fois ?

Objectifs: *À la fin de l'activité, vous serez capable de :*

- *lire et comprendre un algorithme simple ;*
- *suivre l'évolution des variables au cours de son exécution ;*
- *utiliser une condition pour contrôler l'exécution d'un programme ;*
- *combiner plusieurs conditions à l'aide d'opérateurs logiques ;*
- *utiliser une boucle conditionnelle pour répéter des instructions ;*
- *traduire un algorithme simple en Python.*

PARTIE 1: RÉVISION

Bob a écrit l'algorithme suivant. Compléter le tableau des variables afin de suivre l'évolution des valeurs de x, y et z

INSTRUCTIONS	x	y	z
$y \leftarrow 20$	/	20	/
$x \leftarrow y \times 3$	60	20	/
$x \leftarrow x + y$	80	20	/
$z \leftarrow y - 10$	80	20	10
afficher(z)	10.		

PARTIE 2A: DÉCOUVERTE DU TEST CONDITIONNEL "SI… SINON…"

Exemple introductif:

On considère le programme ci-dessous qui permet de calculer le prix (en euros) d'affranchissement d'une lettre en fonction de sa masse (en grammes).

```
masse = float(input("donne-moi un décimal (masse de la lettre en g) : "))

if masse < 30 :
    prix =  1.15
else :
    prix = 1.95

print(prix)
```

1. Quel sera le prix d'affranchissement d'une lettre pesant 40g ? ..1,95..... 10g ? ..1,15..... 30g ? ..1,95.....
2. Que signifie l'instruction **print** ? ..Afficher à l'écran.....
3. Que signifie l'instruction **input** ? ..Demander à l'utilisateur.....
4. Que signifie l'instruction **if** ? ....Si une condition est remplie.....
5. Que signifie l'instruction **else** ? ..Sinon.....
6. *Partie sur ordinateur*
   a. Allumer l'ordinateur et se connecter à son compte
   b. Lancer un navigateur internet et se rendre sur le site Trinket de programmation en Python3 *(le lien est également disponible sur le moodle du cours):* https://trinket.strivemath.org/python3
   c. Saisir le programme ci-dessus dans la fenêtre de gauche en faisant très attention au respect des majuscules et des minuscules, aux deux-points et aux espaces (indentations).
   d. Exécuter ce programme.
   e. Vérifier vos réponses aux questions 1 à 5.

À RETENIR: Syntaxe d'une condition "SI … SINON…" en Python

Syntaxe :	**Exemple :** ␣␣␣␣ **: indentation de 4 espaces**
if *condition* **:** ␣␣␣␣ instructions 1 **else :** ␣␣␣␣ instructions 2	**if** *moyenne* < 10 **:** ␣␣␣␣ print("Ta moyenne est insuffisante.") ␣␣␣␣ print("Tu peux mieux faire.") **else :** ␣␣␣␣ print("Ta moyenne est satisfaisante.")

Attention : en Python, l'indentation permet d'indiquer quelles instructions appartiennent au *if* ou au *else*.

Exercice 1 : Entier supérieur à 33

Compléter ci-dessous un programme en Python qui demande de saisir un entier *n* et affiche suivant le cas : « **l'entier dépasse 33** » ou « **l'entier ne dépasse pas 33** »

```
n  = int(input("entrez un entier n:"))
if  n > 33.    :
    print("..l'entier...dépasse.33.................")
else :
    print (l'entier ne dépasse pas 33")
```

Partie sur ordinateur:

a. Saisir le programme dans Trinket.
b. Exécuter ce programme, vérifier qu'il fonctionne et le tester pour *n=33, n=34* et *n=20*.

Exercice 2 : Prix d'une place de cinéma

Les tarifs d'entrée au cinéma sont :
• avant 10 ans : 5 € • de 10 à 17 ans : 8 € • adultes : 10 €

1. Compléter le programme en Python suivant afin qu'il demande de saisir un entier *age* et affiche, suivant le cas, le prix à payer)

```
age = int(input("entrez votre âge : "))
if  age  < 10..   :
    prix = .5.......
else :
    if ..age..<18..... :
        prix = ..8.......
    else :
        prix = ..10.......
print("prix à payer =",prix)
```

3. *Partie sur ordinateur*

a. Saisir le programme dans Trinket.
b. Exécuter ce programme, vérifier qu'il fonctionne et qu'il répond correctement au problème.

APPEL → **Appeler le professeur pour vérification**

PARTIE 2B: PERFECTIONNEMENT: COMBINER ET COMPARER DES CONDITIONS

Exemple :

1. *Partie sur ordinateur*
   a. Dans Trinket, saisir tour à tour chacun des programmes suivants, les exécuter et recopier les affichages obtenus.

```
a = 5
b = 10
if a==5 and b==10 :
    print("OUI")
else :
    print("NON")
```

Affichage : ..OUI.....

```
a = 8
b = 10
if a==5 and b==10 :
    print("OUI")
else :
    print("NON")
```

Affichage : ..NON.....

```
a = 8
b = 10
if a==5 or b==10 :
    print("OUI")
else :
    print("NON")
```

Affichage : ..OUI.....

```
a = 8
b = 10
if a!=5 and b==10 :
    print("OUI")
else :
    print("NON")
```

Affichage : ..OUI.....

2. D'après ce qui précède, que signifient les mots clés suivants :

   and ...et..... **or** ...ou.....

   == ...est égal à... **!=** ...est différent de

3. Compléter alors le programme suivant pour qu'il affiche OUI uniquement si **a** vaut 10 ou **b** ne vaut pas 11.

```
a = ..10...
b = ..11...
if ..a == 10.....ou...b != 11..................  :
    print("OUI")
else :
    print("NON")
```

4. *Partie sur ordinateur*
   a. Saisir votre programme de la question 3 dans Trinket.
   b. Exécuter ce programme, vérifier qu'il fonctionne et qu'il répond correctement au problème.

APPEL → **Appeler le professeur pour vérification**

PARTIE 3: DÉCOUVERTE DE LA BOUCLE "TANT QUE..."

Exemple introductif

On considère l'algorithme ci-dessous ainsi que sa traduction en Python.
Les deux instructions de la boucle sont réalisées indéfiniment, tant que la condition `entier⩽499` est vraie.

```
entier ← 5
tant_que  (entier ⩽ 499)  faire:
    entier ← entier × 10
    afficher(entier)
afficher("fin")
```

↻ boucle

```
entier = 5
while   entier <= 499 :
    entier = entier * 10
    print(entier)
print("fin")
```

↻ boucle

1. Que signifie *while* en Python ? ...Tant que la condition est vérifiée.....
2. Quelle condition doit rester vraie pour que les instructions de la boucle continuent à être exécutées ?
   ...la variable "entier" doit être inférieure ou égale à 499.....
3. Sans exécuter le programme, déterminer ce qui sera affiché à l'écran parmi les propositions ci-dessous.

5 50 500 **fin**	50 500 **fin**	50 500 5000 **fin**	50 **fin**

4. *Partie sur ordinateur*
   a. Saisir le programme ci-dessus dans Trinket.
   b. Exécuter ce programme, vérifier qu'il fonctionne et vérifier votre réponse à la question 2.
5. Pourquoi la boucle finit-elle par s'arrêter ? ...Parce que "entier = 500" et donc la condition "entier ⩽ 499" est fausse.....

À RETENIR: Syntaxe d'un "TANT QUE" en Python

Les instructions indentées sont répétées tant que la condition est vraie.

Syntaxe :	**Exemple :**	␣␣␣␣ **: indentation de 4 espaces**
while *condition* **:** ␣␣␣␣ instruction 1 ␣␣␣␣ instruction 2 ␣␣␣␣ ...	**while** $x > y$ **:** ␣␣␣␣ $a = x$ ␣␣␣␣ $b = x+1$ ␣␣␣␣ $c = y+8$	ne pas oublier les **:**

Exercice 1:

1. Compléter les pointillés afin de traduire l'algorithme suivant en Python

Algorithme	**Code Python associé**
afficher(« bonjour »)	**print**("bonjour")
$a \leftarrow 10$	$a = 10$
tant que $a < 1000$ **:**	while a<1000 :
$a \leftarrow a+2$	a = a + 2
$a \leftarrow a \times 10$	a = a x 10
afficher("a=", a)	print("a=", a)
afficher(« fin »)	print ("fin")

2. Un élève a écrit le programme précédent puis l'exécute.
   Parmi les propositions suivantes, laquelle s'affiche à l'écran ?

bonjour a=12 **fin**	**bonjour** a=120 **fin**	**bonjour** a=122 **fin**	**bonjour** a=1200 **fin**	**bonjour** a=1220 **fin**	**bonjour** a=12200 **fin**

3. *Partie sur ordinateur*
   a. Saisir le programme ci-dessus dans Trinket.
   b. Exécuter ce programme, vérifier qu'il fonctionne et vérifier votre réponse à la question 2.

APPEL → **Appeler le professeur pour vérification**

Exercice 2:

Le programme suivant permet de déterminer le plus petit entier n dont le triple dépasse 10000.

```
1 n = 0
2 while n*3  <= 10000 :
3     n = n +1
4 print("n =",n)
```

1. Modifier ci-dessous la ligne 2 du programme précédent afin de déterminer le plus petit entier n dont le double dépasse 50000.

```
2 while   .n*.2...<.=..50.000.....   :
```

3. *Partie sur ordinateur*
   a. Saisir votre programme dans Trinket.
   b. Exécuter ce programme, vérifier qu'il fonctionne et qu'il répond correctement au problème.

<u>BILAN DE L'ACTIVITÉ 1.2:</u> Conditions et boucles en Python

Dans un programme, les instructions ne sont pas toujours exécutées les unes après les autres. Une condition permet d'exécuter des instructions différentes selon qu'une situation est vérifiée ou non. Une boucle permet de répéter plusieurs fois les mêmes instructions tant qu'une condition reste vérifiée. Ces structures permettent ainsi de contrôler l'ordre et la répétition des instructions d'un programme.

Compléter le tableau puis mémoriser les principaux mots-clés et leur syntaxe en Python.

Langage naturel :	**Python :**	**Exemple :**
variable ← valeur	*variable* = valeur	$x = 10.30$
afficher (*n*)	print (*n*)	**print**(*n*)
demander(*ch*) *ch* chaine de caractères	*ch* = **input**("*ch*="))	*nom* = **input**("*ton nom* ? :")
demander(*n*) *n* entier	*n* = int (**input**("*n*="))	*age* = **int**(**input**("*age*="))
demander(*x*) *x* décimal	*t* = float (**input**("*x*="))	*t* = **float** (**input**("*taille*="))
si (*condition*) : instructions 1	if ... (*condition*) : ␣␣␣␣ instructions 1	**if** $(n > 100)$: ␣␣␣␣ $s = s + 1$
sinon : instructions 2	else : ␣␣␣␣ instructions 2	**else :** ␣␣␣␣ $s = s - 1$ ␣␣␣␣ $n = n + 3$
tant que (*condition*) instructions	while (*condition*) : ␣␣␣␣ instructions	**while** $(n > 100)$: ␣␣␣␣ $s = s + n$ ␣␣␣␣ $n = n - 8$

<u>Pour aller plus loin :</u> opérateurs logiques et opérateurs de comparaison

Opérations logiques

```
b1 or b2    # b1 ou b2
b1 and b2   # b1 et b2
not b1      # contraire de b1
```

Comparaison entre deux valeurs

```
x < y    # x inférieur à  y
x > y    # x supérieur  à  y
x == y   # x égal  à  y
x != y   # x différent de y
x <= y   # x inférieur ou égal  à  y
x >= y   # x supérieur ou égal  à  y
```