

Master 1 Humanités numériques – Algorithmique et programmation

TD 3 : Boucles

J. Darmont – <http://eric.univ-lyon2.fr/~jdarmont/>

Les exercices ci-dessous sont à formuler en langage algorithmique textuel, en Blockly, puis en Python. Tester l'exécution dans ces deux derniers cas.

Exercice 1

1. Définir une constante TAILLE à une valeur quelconque (pas trop grande avec Blockly).
2. Afficher une ligne verticale composée d'autant de caractères * que la TAILLE.

*
*
*
*
*

Exercice 2

1. Reprendre l'exercice 1.
2. Au lieu d'afficher une ligne verticale, afficher un triangle rectangle composé de caractères *.

```
*  
**  
***  
****  
*****
```

Indice : au lieu d'afficher un caractère * à chaque ligne, ajouter à chaque fois le caractère * à une variable ligne (concaténation de chaînes de caractères) et afficher cette ligne, qui grandira au fur et à mesure.

Exercice 3

1. Lors d'un événement sportif de type course (à pied, en vélo, natation...), on souhaite saisir par ordre d'arrivée le numéro de dossard de chaque concurrent-e, ainsi que son temps.
2. Le processus doit continuer jusqu'à ce que l'on saisisse le numéro de dossard -1.

Exercice 4

Reprendre l'exercice 3. Transformer l'algorithme pour éviter d'avoir à saisir un temps lorsque la saisie s'arrête (dossard -1).

Exercice 5

Comme les tables de division sont au mieux peu courantes, programmons-les !

1. Définir une constante TAILLE à une valeur quelconque (pas trop grande avec Blockly).
2. Il s'agit de calculer toutes les combinaisons possibles de divisions de $1 \div 1$ à $\text{TAILLE} \div \text{TAILLE}$. Pour cela, il faut imbriquer deux boucles. Exemple de résultat avec $\text{TAILLE} = 3$:

$$1 \div 1 = 1.0$$

$$1 \div 2 = 0.5$$

$$1 \div 3 = 0.3333333333333333$$

$$2 \div 1 = 2.0$$

$$2 \div 2 = 1.0$$

$$2 \div 3 = 0.6666666666666666$$

$$3 \div 1 = 3.0$$

$$3 \div 2 = 1.5$$

$$3 \div 3 = 1.0$$

Correction Exercice 1

Algorithme ligneEtoiles

Const TAILLE ← 5

Var i : Entier

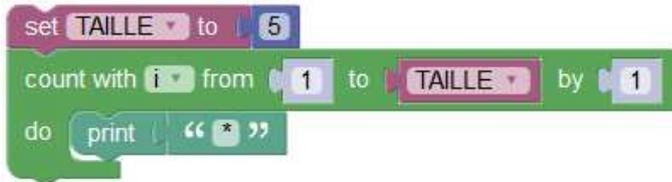
Début

Pour i de 1 à TAILLE faire

Écrire("*")

Fin pour

Fin



```
TAILLE = 5
```

```
for i in range(1, TAILLE + 1):
```

```
    print("*")
```

Correction Exercice 2

Algorithme triangleEtoiles

Const TAILLE ← 5

Var i : Entier

Var ligne : Chaîne

Début

ligne ← "" {Au début, la ligne est vide}

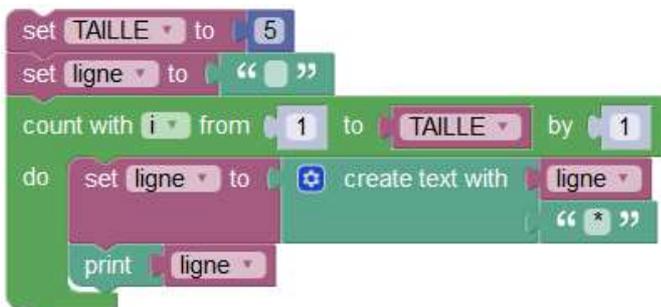
Pour i de 1 à TAILLE faire

ligne ← ligne | "*" {On ajoute une étoile à la ligne}

Écrire("*")

Fin pour

Fin



```

TAILLE = 5
ligne = "" # Au début, la ligne est vide
for i in range(1, TAILLE + 1):
    ligne += "*" # On ajoute une étoile à la ligne
    print(ligne)

```

Correction Exercice 3

Algorithme saisieTemps

Var dossard : Entier

Var temps : Réel

Début

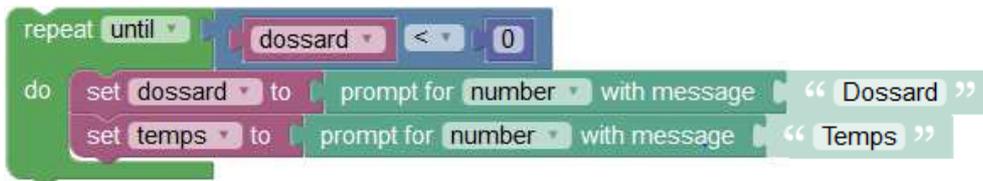
Répéter

Lire("Dossard : ", dossard)

Lire("Temps : ", temps)

Jusqu'à dossard < 0

Fin



```

while True: # à éviter !
    dossard = int(input("Dossard : "))
    temps = float(input("Temps : "))
    if dossard < 0:
        break # moche !

```

Correction Exercice 4

Algorithme saisieTemps2

Var dossard : Entier

Var temps : Réel

Début

Lire("Dossard : ", dossard) {Lecture du premier dossard}

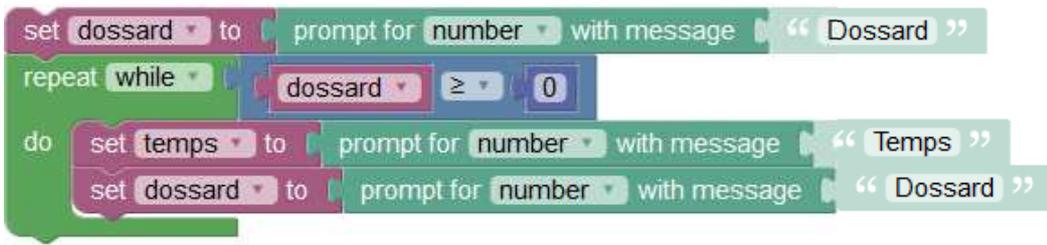
Tant que dossard ≥ 0 faire

 Lire("Temps : ", temps)

 Lire("Dossard : ", dossard) {Lecture du dossard suivant}

Fin tant que

Fin



```
dossard = int(input("Dossard : "))           # Lecture du premier dossard
while dossard >= 0:
    temps = float(input("Temps : "))
    dossard = int(input("Dossard : "))      # Lecture du dossard suivant
```

Correction Exercice 5

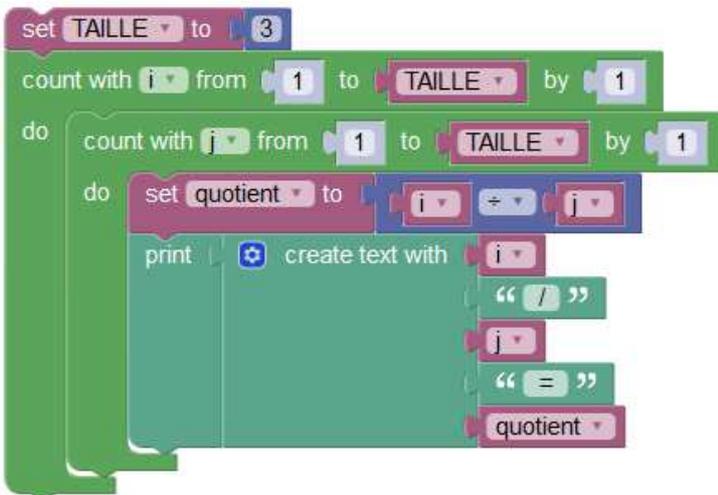
Algorithme tableDiv

Const TAILLE ← 3
 Var i, j : Entier
 Var quotient : Réel

Début

Pour i de 1 à TAILLE faire
 Pour j de 1 à TAILLE faire
 quotient ← i ÷ j
 Écrire(i, " ÷ ", j, " = ", quotient)
 Fin pour
 Fin pour

Fin



```
TAILLE = 3
for i in range(1, TAILLE + 1):
    for j in range(1, TAILLE + 1):
        quotient = i / j
        print(i, "/", j, "=", quotient)
```