Tables de Pythagore de n à m [lp04] - Exercice

Karine Zampieri, Stéphane Rivière, Béatrice Amerein-Soltner

Unisciel algoprog Université Haute-Alsace

Version 17 mai 2018

Table des matières

Python - Tables de Pythagore de n à m (Solution)

Mots-Clés Structures répétitives ■
Requis Structures de base, Structures répétitives, Algorithmes paramétrés ■
Difficulté • ○ ○

Objectif
Cet exercice affiche la partie triangulaire inférieure des tables de PYTHAGORE des entiers de n à m compris dans [1..15].

...(énoncé page suivante)...

1 Tables de Pythagore de n à m / pgtpythagore

1.1 Stratégie de résolution

Voici un extrait du résultat attendu :

```
Deux numeros de table dans [1..15]? 6 11
   1   2   3   4   5   6   7   8   9  10
   6  12  18  24  30  36  42  48  54  60
   7  14  21  28  35  42  49  56  63  70
 ...
  11  22  33  44  55  66  77  88  99 110
```

Comment pouvez-vous procéder pour afficher ces tables ?

Solution simple
Une première solution est d'utiliser deux structures `Pour` imbriquées : l'une concerne les lignes `j` de la table, l'autre les colonnes `k`. L'instruction à l'intérieur de l'itérative imbriquée est une instruction d'affichage du produit `j*k`. L'instruction supplémentaire au niveau de l'itérative `j` est l'instruction pour passer à la ligne suivante.

Une autre solution est de définir une procédure `afficherTM(n)` qui affiche la table de multiplication d'un entier `n` : elle utilise une structure `Pour` qui concerne les colonnes `k` de la table. Il suffit alors d'une seule structure `Pour` qui concerne les lignes `j` de la table.

Nous allons programmer cette deuxième solution.

Comment vérifier que les deux numéros de table sont bien dans l'intervalle spécifié ?

Solution simple
En réalisant une saisie validée des numéros de table.

1.2 Procédure afficherTM

Écrivez le **profil** d'une procédure `afficherTM(n)` qui affiche la table de Pythagore d'un entier `n`.

Solution Paramètres
Entrants : Un entier `n`

Écrivez son corps.
Formatez les entiers sur 4 positions.
Exemple : Table de l'entier 7.

```
   7  14  21  28  35  42  49  56  63  70
```

Validez votre procédure avec la solution.

Solution Python

```
def afficherTM(n):
    """  Affiche la table de Pythagore d'un entier

     :param n: un entier
    """
    for k in range(1, 10+1):
        print("{:4d}".format((k * n)), end="")
    print('\n', end="")
```

1.3 Procédure saisirNumerosT

Écrivez une fonction `dansIntervalle(n,a,b)` qui teste et renvoie `Vrai` si un entier `n` est dans l'intervalle `[a..b]` d'entiers, `Faux` sinon.

Validez votre fonction avec la solution.

Solution Python

```
def dansIntervalle(n, a, b):
    """  Prédicat d'un entier dans un intervalle

     :param n: un entier
     :param a: un entier
     :param b: un entier
     :return: Vrai si n est dans [a..b] (a <= b), Faux sinon
    """
    return ((a <= n) and (n <= b))
```

Écrivez le **profil** d'une procédure `saisirNumerosT(n1,n2)` qui saisit deux numéros de table dans `n1` (entier) et dans `n2` (entier).

Solution Paramètres

Sortants : Deux entiers `n1` et `n2`

Écrivez son corps de sorte que les deux entiers soient compris dans [1..15] avec `n1` inférieur ou égal à `n2`. Affichez l'invite :

```
Deux numeros de table dans [1..15]?
```

Aide méthodologique

Utilisez une structure répétitive (`Répéter` ou `TantQue`) qui saisit `n1` et `n2` jusqu'à ce que les conditions soient réalisées. Si vous utilisez un `TantQue`, n'oubliez pas de faire une saisie des entiers avant la structure `TantQue` de sorte à ce que la condition ait un sens.

Validez votre procédure avec la solution.

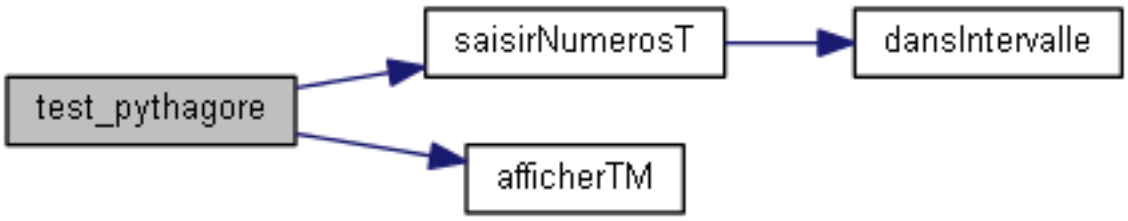
Solution Python

```
def saisirNumerosT():
    """  Saisie des numéros de table dans [1..15]

     :return: le tuple (n1,n2) avec n1 <= n2
    """
    n1, n2 = 0, 0
    while not (dansIntervalle(n1, 1, 15) and dansIntervalle(n2, 1, 15) and n1 <= n2):
        print("Deux numeros de table dans [1..15]? ", sep="", end="")
        n1 = int(input())
        n2 = int(input())
    return (n1, n2)
```

1.4 Tables de Pythagore de n à m

Ce problème affiche les tables de Pythagore des entiers de n à m compris dans [1..15]. Il utilise les procédures `saisirNumerosT` et `afficherTM`.



Écrivez une procédure `test_pythagore` qui demande deux numéros de tables dans `n1` (entier) et dans `n2` (entier) tous deux compris dans [1..15] par appel à la procédure `saisirNumerosT`.

Affichez les tables de Pythagore pour tous les entiers `n` de `n1` à `n2` par appel à la procédure `afficherTM`. Ajoutez l'en-tête `1 2 ... 10` (si `n1` est différent de 1).

Aide méthodologique

Utilisez une boucle `Pour` de compteur `n` (entier) variant de `n1` à `n2`.

Testez. Exemple d'exécution :

```
Deux numeros de table dans [1..15]? 6 11
   1   2   3   4   5   6   7   8   9  10
   6  12  18  24  30  36  42  48  54  60
   7  14  21  28  35  42  49  56  63  70
   8  16  24  32  40  48  56  64  72  80
   9  18  27  36  45  54  63  72  81  90
  10  20  30  40  50  60  70  80  90 100
  11  22  33  44  55  66  77  88  99 110
```

Validez votre procédure avec la solution.

Solution Python @[pgtpythagore.py]

```
def test_pythagore():
    """ @test """
    n1, n2 = saisirNumerosT()
    if n1 != 1:
        afficherTM(1)
    for n in range(n1, n2+1):
        afficherTM(n)
```

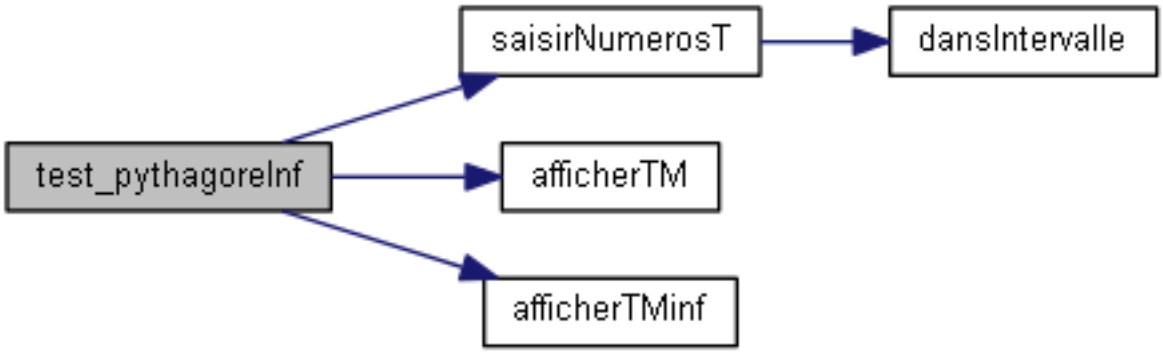
1.5 Partie inférieure des tables

Ce problème affiche la partie triangulaire inférieure des tables de PYTHAGORE de n à m donnés. Voici un extrait du résultat attendu.

```
Deux numeros de table dans [1..15]? 7 13
  1   2   3   4   5   6   7   8   9  10
  7  14  21  28  35  42  49
  8  16  24  32  40  48  56  64
  ...
  13  26  39  52  65  78  91 104 117 130
```

Copiez/collez la procédure `afficherTM` en la procédure `afficherTMinf(n)` puis modifiez la boucle sur `k` de 1 à `n` compris (et non pas 10 pour la table complète) de sorte à afficher la table de PYTHAGORE inférieure de `n`. **Attention,** l'indice maximal de la colonne est 10.

Copiez/collez la procédure `test_pythagore` en la procédure `test_pythagoreInf` puis modifiez l'appel de la procédure pour afficher la partie triangulaire inférieure des tables de PYTHAGORE.



Testez. Exemple d'exécution :

```
Deux numeros de table dans [1..15]? 7 13
   1   2   3   4   5   6   7   8   9  10
   7  14  21  28  35  42  49
   8  16  24  32  40  48  56  64
   9  18  27  36  45  54  63  72  81
  10  20  30  40  50  60  70  80  90 100
  11  22  33  44  55  66  77  88  99 110
  12  24  36  48  60  72  84  96 108 120
  13  26  39  52  65  78  91 104 117 130
```

Validez vos procédures avec la solution.

Solution Python @[pgtpythagore.py]

```
def afficherTMinf(n):
    """  Affiche la table de Pythagore inférieure d'un entier

     :param n: table de n
    """
    bsup = (n if n <= 10 else 10)
    for k in range(1, bsup+1):
        print("{:4d}".format((k * n)), end="")
    print('\n', end="")
```

```
def test_pythagoreInf():
    """ @test """
    n1, n2 = saisirNumerosT()
    if n1 != 1:
        afficherTM(1)
    for n in range(n1, n2+1):
        afficherTMinf(n)
```

2 Références générales

Comprend [Lery-AL1 :c2 :ex1] ■