Nom :	Prénom :	Groupe :

Durée : 90' Documents, calculettes, téléphones et ordinateurs interdits

1 Calcul de π (1)

Définir une fonction qui calcule π à l'ordre n selon la formule :

$$\frac{\pi^2}{6} = 1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16} - \cdots + \frac{1}{n^2} = \sum_{k=1}^{n} \frac{1}{k^2}$$

2 Conversion base $b \to$ décimal

Définir une fonction qui calcule la valeur décimale n d'un entier positif t codé en base b.

Exemples : `b = 2  t = [0, 0, 0, 1, 0, 1, 1, 1]` $\to$ `n = 23`
`b = 5  t = [0, 0, 4, 3]` $\to$ `n = 23`
`b = 21 t = [1, 2]` $\to$ `n = 23`
`b = 25 t = [0, 0, 0, 0, 0, 23]` $\to$ `n = 23`

3 Courbes fractales

On considère la procédure `p` ci-contre :

1. On considère l'appel `p(1,300)` et le crayon initialement en (0,0) avec une direction de -90 (vers le bas). Dessiner le résultat de cet appel.
2. On considère l'appel `p(3,300)` et le crayon initialement en (0,0) avec une direction de -90 (vers le bas). Dessiner le résultat de cet appel.

```
def p(n,d):
    assert type(n) is int
    assert n >= 0
    if n == 0: forward(d)
    else:
        p(n-1,d/3.)
        right(60)
        p(n-1,d/3.)
        left(120)
        p(n-1,d/3.)
        right(60)
        p(n-1,d/3.)
    return
```

4 Portée des variables

On considère les fonctions `f`, `g` et `h` suivantes :

```
def f(x):
    x = 3*x
    print('f', x)
    return x
```

```
def g(x):
    x = 3*f(x)
    print('g', x)
    return x
```

```
def h(x):
    x = 3*g(f(x))
    print('h', x)
    return x
```

Qu'affichent les appels suivants ?

1.
```
>>> x = 2
>>> print(x)

>>> y = f(x)
>>> print(x)

>>> z = g(x)
>>> print(x)

>>> t = h(x)
>>> print(x)
```

1.
```
>>> x = 2
>>> print(x)

>>> x = f(x)
>>> print(x)

>>> x = g(x)
>>> print(x)

>>> x = h(x)
>>> print(x)
```

5 Calcul de π (2)

On considère la fonction `g` ci-contre :

1. Calculer toutes les valeurs possibles de $g(n, m)$ pour $n \in [0, 6]$.
2. Vérifier que `12.*g(5,5)/g(6,6)` est une bonne approximation de π.

```
#--------------------------------
def g(n,m):
#--------------------------------
    assert type(n) is int
    assert type(m) is int
    assert 0 <= m and m <= n
    if n == 0 and m == 0:
        c = 1
    else:
        if m == 0: c = 0
        else:
            c = 0
            for i in range(1,m+1):
                c = c + g(n-1,n-i)
    return c
#--------------------------------
```

$g(n,m)$	$m=0$	$m=1$	$m=2$	$m=3$	$m=4$	$m=5$	$m=6$
$n=0$							
$n=1$							
$n=2$							
$n=3$							
$n=4$							
$n=5$							
$n=6$							

$$12 \cdot \frac{g(5,5)}{g(6,6)} = 12 \cdot \frac{\quad}{\quad} =$$