exofonctions

October 15, 2024

1 Exercices fonctions Première - NSI

1.1 Exercice 1 :

Écrire la fonction **est_pair(x : int) → bool** qui renvoie **True** si l'entier **x** est pair, **False** sinon.

1.2 Exercice 2 :

Écrire la fonction **valeur_absolue(x: int) → int** qui renvoie la valeur absolue de l'entier **x**.

1.3 Exercice 3 :

Écrire la fonction **surface(r: float) → float** qui renvoie l'aire d'un cercle de rayon **r**. Le type **float** représente un nombre réel.

1.4 Exercice 4 :

Écrire la fonction **est_majeur(age: int) → bool** qui renvoie **True** si la personne d'âge **age** est majeure, **False** sinon.

1.5 Exercice 5 :

Écrire la fonction **puissance(x: int, n: int) → int** qui renvoie **x** à la puissance **n**. Pour rappel xn=x ×x× ….×x n. On utilisera une boucle pour effectuer le calcul.

1.6 Exercice 6 :

Écrire la fonction **lancer_des() → int** qui simule le lancer de deux dés à six faces et renvoie la somme des valeurs obtenues.

1.7 Exercice 7 :

Écrire la fonction **pythagore(a: int, b: int, c: int) → bool** qui renvoie **True** si le triangle formé par les côtés de mesures **a b c** est rectangle. On supposera que les mesures sont des entiers donnés dans l'ordre croissant.

1.8 Exercice 8 :

Écrire la fonction **somme(n: int) → int** qui renvoie la somme des entiers de 1 à n.

1.9 Exercice 9 :

Écrire la fonction **est_premier(n: int) → bool** qui renvoie **True** si **n** est premier.

1.10 Exercice 10 :

Le calcul $0,5 \times (x + \frac{a}{x})$ permet d'effectuer une bonne approximation de $\sqrt{a}$ si on choisit une valeur de x pas trop éloignée de $\sqrt{a}$.Par exemple, pour $\sqrt{50}$ on choisit $x = 7$.
- $0,5 \times (7 + \frac{50}{7}) \simeq 7,0714$ - $0,5 \times (7,0714 + \frac{50}{7,0714}) \simeq 7,0711$ - … - après 20 itérations, on trouve : 7,0710678118654755

Et la calculatrice donne une valeur $\sqrt{50} = 7,0710678118654755$

1. Écrire la fonction **valeur_proche(a: int) → int** qui renvoie l'entier inférieur le plus proche de $\sqrt{a}$ Par exemple, l'appel valeur_proche(50) doit renvoyer 7. Pour trouver ce nombre il faut remarquer que $7^2 = 49 < 50$.
2. Écrire la fonction **racine(a: int) → float** qui renvoie la racine carrée de **a**. La fonction utilisera la formule présentée en début d'exercice.

1.11 Exercice 11 :

Turtle

1. Écrire une fonction **triangle(c)** qui trace un triangle de côté **c**.
2. Écrire le programme qui affiche la figure1.

Sapin

1.12 Évaluer le résultat de l'appel d'une fonction avec des paramètres donnés

Quelle est la valeur à l'évaluation de l'expression … avec la définition de fonction Python suivante ?

1.12.1 Exercice 1

Quelle est la valeur à l'évaluation de l'expression : f(-14) avec la définition de fonction Python suivante ?

```
def f(x):
    if x > 0:
        return (x)
    else:
        return (-x)
```

1.12.2 Exercice 2

Quelle est la valeur à l'évaluation de l'expression : g(5,3) avec la définition de fonction Python suivante ?

```
def g(a,b):
    r = 1
    for i in range(b):
        r = r * a
    return (r)
```

1.12.3 Exercice 3

Quelle est la valeur à l'évaluation de l'expression : max(max(42, 36), max(53, 9)) avec la définition de fonction Python suivante ?

```
def max(a,b):
    if a > b :
        m = a
    else :
        m = b
    return (m)
```

1.13 Modéliser un traitement par une fonction en spécifiant ses paramètres

1.13.1 Exercice 1

Spécifier le nom, les paramètres d'entrée et le type de résultat à renvoyer pour une fonction destinée à calculer…

1.13.2 Exercice 2

Spécifier le nom, les paramètres d'entrée et le type de résultat à renvoyer pour une fonction destinée à calculer la puissance nieme d'un nombre entier.

1.13.3 Exercice 3

Spécifier le nom, les paramètres d'entrée et le type de résultat à renvoyer pour une fonction destinée à vérifier si un mot est un palindrome (exemple : rever, laval…).

1.14 Anticiper l'écriture du corps d'une fonction

Écrire une fonction Python… , qui étant donnés… , renvoie…

1.14.1 Exercice 1

Écrire une fonction Python `max3` qui étant donnés trois entiers `a` , `b` et `c` renvoie le maximum des trois.

On peut avoir en particulier l'exécution suivante :

max3 (24, 37, 12) 37

1.14.2 Exercice 2

Ecrire une fonction Python `puissance`, qui étant donnés deux nombres entiers `x` et `n`, renvoie la valeur de `x` à la puissance `n`.

On peut avoir en particulier l'exécution suivante :

puissance (5, 3) 125

1.14.3 Exercice 3

Ecrire une fonction Python `nombre_de_a`, qui étant donnée une chaîne de caractères, renvoie le nombre de 'a' qu'elle contient.

On peut avoir en particulier l'exécution suivante :

nombre_de_a ("abracadabra") 5

1.15 Décomposer un traitement complexe en utilisant la composition de fonctions

1.15.1 Exercice 1

Décomposer le problème suivant en identifiant deux parties pouvant chacune être programmées par une fonction. Ecrire une fonction python `verification` permettant de vérifier qu'une méthode de chiffrement / déchiffrement de caractères fonctionne bien, c'est à dire qu'en déchiffrant le caractère chiffré on retrouve bien le caractère d'origine. On peut spécifier des fonctions intermédiaires, sans chercher à les écrire.

```
def verification():
    for car in "abcdefghijklmnopqrstuvwxyz":
```

1.15.2 Exercice 2

Décomposer le problème suivant en identifiant deux parties pouvant chacune être programmées par une fonction.

Ecrire une fonction python `nb0fact` permettant de calculer pour un nombre `n` donné, le nombre de 0 à droite dans l'écriture en base 10 de `n!`. On peut spécifier des fonctions intermédiaires, sans chercher à les écrire.

[ ]:

1.16 Généraliser un traitement, par la définition d'une fonction avec paramètres ou par l'ajout d'un paramètre à une fonction

Dans le programme Python suivant, remplacer les instructions écrites plusieurs fois, en définissant et utilisant une fonction

1.16.1 Exercice 1

Dans le programme suivant, remplacer les instructions écrites plusieurs fois, en définissant et utilisant une fonction, tout en conservant les mêmes affichages.

```
a = 4
b = 9
if a == 0 :
    print ("L'équation :", a, "x +", b, "= 0", "n'a pas de solution")
else :
    print ("L'équation :", a, "x +", b, "= 0", "a une solution :", -b/a)
a = 2
b = -8
if a == 0 :
    print ("L'équation :", a, "x +", b, "= 0", "n'a pas de solution")
else :
    print ("L'équation :", a, "x +", b, "= 0", "a une solution :", -b/a)
```

[ ]:

1.16.2 Exercice 2

Dans le programme suivant, remplacer les instructions écrites plusieurs fois, en définissant et utilisant une fonction, tout en conservant le même résultat dans la variable m.

```
a = 42
b = 53
c = 56
d = 31
if a > b :
    e = a
else :
    e = b
if c > d :
    f = c
else :
    f = d
if e > f :
    m = e
else :
    m = f
```

[ ]:

1.17 Abstraire un traitement par une définition de fonction avec paramètres et commentaire

1.17.1 Exercice 1

Décrire le traitement effectué par la fonction suivante en faisant abstraction de la méthode utilisée.

```
def puissance(a,b):
    r = 1
    for i in range(b):
        r = r * a
    return (r)
```

1.17.2 Exercice 2

Décrire le traitement effectué par la fonction suivante en faisant abstraction de la méthode utilisée.

```
def nombre_de_dans(car, mot):
    n = 0
    for c in mot :
        if c == car :
            n = n + 1
    return(n)
```