Fonctions

Cours 1 : Déclarations

Exercice 1 : IMC
version simple en python

```python
def imc(m, t):
    res = m / t **2
    if res < 18.5:
        return 0 # sous-poids
    elif res > 25:
        return 2 # sur-poids
    else :
        return 1 # normal
```

Exercice 1 : IMC
version simple
en java,
C et python

```java
public int imc(double m, double t) {
    double res = m / Math.sqrt(t);
    if(res < 18.5)
        return 0; // sous-poids
    else if(res > 25)
        return 2;
    else return 1;
}
```

```c
int imc(float m, float t)
{
    float res = m / pow(t, 2);
    if(res < 18.5)
        return 0; // sous-poids
    else if(res > 25)
        return 2; // sur-poids
    else
        return 1; // normal
}
```

```python
def imc(m, t):
    res = m / t **2
    if res < 18.5:
        return 0 # sous-poids
    elif res > 25:
        return 2 # sur-poids
    else :
        return 1 # normal
```

Exercice 1 : IMC

version simple en C

fonctions__exercice1.c

```c
#include <stdio.h>
#include <math.h>

/* Renvoie la classe d'Indice de Masse Corporelle
selon la formule IMC = masse / taille²
    Paramètres :
        m : float (masse en kg)
        t : float (taille en mètres)
    Préconditions :
        m > 0
        t > 0
    Retourne :
        int : 0, 1 ou 2 selon la classe
        (sous-poids / normal / surpoids) */
int imc(float m, float t)
{
    float res = m / pow(t, 2);
    if(res < 18.5)
        return 0; // sous-poids
    else if(res > 25)
        return 2; // sur-poids
    else
        return 1; // normal
}


int main(void)
{
    printf("L'IMC est de classe : %d", imc(70, 1.8));
    return 0;
}
```

Exercice 1 : IMC

version simple en java

fonctions__exercice1.java

```java
public class Exercice1 {
    /*
    Renvoie la classe d'Indice de Masse Corporelle
    selon la formule IMC = masse / taille²
    Paramètres :
        m : double (masse en kg)
        t : double (taille en mètres)
    Préconditions :
        m > 0
        t > 0
    Retourne :
        int : 0, 1 ou 2 selon la classe
        (sous-poids / normal / surpoids)
  */
  public int imc(double m, double t) {
    double res = m / Math.sqrt(t);
    if(res < 18.5)
      return 0; // sous-poids
    else if(res > 25)
      return 2;
    else return 1;
  }

  public static void main(String args[]) {
  }
}
```

Exercice 1 : IMC

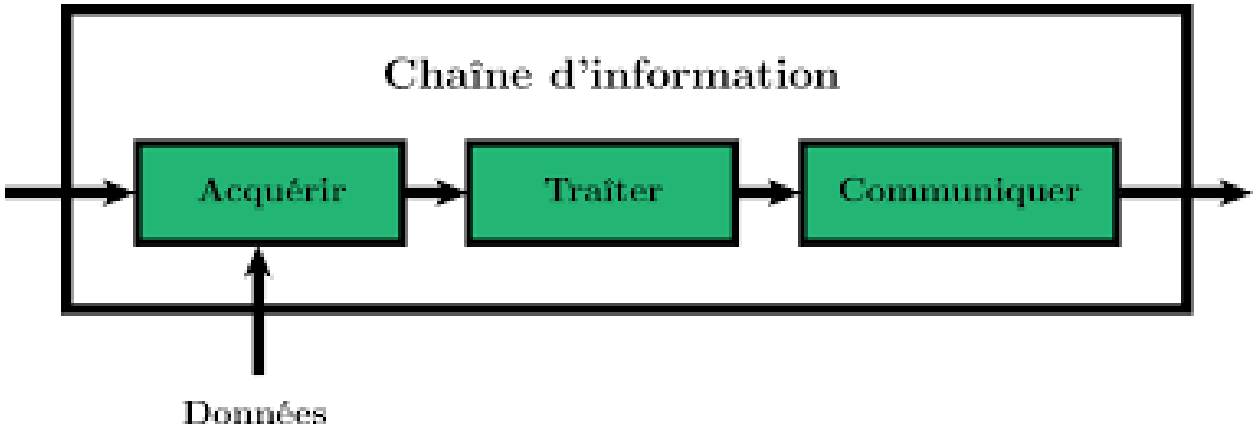
depuis python 3.5 : annotations de types (optionnelle)

fonctions__exercice1.py

```python
# Fonctions - EXERCICE 1

def imc(m: float, t: float) -> int:
    res = m / t **2
    if res < 18.5:
        return 0 # sous-poids
    elif res > 25:
        return 2 # sur-poids
    else :
        return 1 # normal
```

La chaîne d'information



ENTRÉES : paramètres	→	INTÉRIEUR DE LA FONCTION : traitement	→	SORTIE : résultat

Exercice 1 : IMC
version complète

Description
de la fonction

Définition
de la fonction

Test
de la fonction

fonctions_exercice1.py

```python
# Fonctions - EXERCICE 1

def imc(m: float, t: float) -> int:
    """
    Renvoie la classe d'Indice de Masse Corporelle
    selon la formule IMC = masse / taille²
    Paramètres :
        m : float (masse en kg)
        t : float (taille en mètres)
    Préconditions :
        m > 0
        t > 0
    Retourne :
        int : 0, 1 ou 2 selon la classe
        (sous-poids / normal / surpoids)
    """
    assert(m > 0)
    assert(t > 0)
    res = m / t **2
    if res < 18.5:
        return 0 # sous-poids
    elif res > 25:
        return 2 # sur-poids
    else :
        return 1 # normal

# Tests unitaires
assert(imc(50, 1.7) == 0)
assert(imc(60, 1.65) == 1)
assert(imc(70, 1.6) == 2)
```

Déclaration

Documente
la fonction
`imc.__doc__`

Assertions (= vérifications)

Calcul

Renvoi du résultat

Commenter ?
Quoi ?

Source :
https://lesjoiesducode.fr/?s=commentaire

Com
Quo

https:
e.fr/?

Vois-tu petit, du bon code, ça n'a pas besoin de commentaire

Les Joies du Code, commit du 29 Sep 2020

elier 2023

Pourquoi commenter le code?

- Parce qu'un code doit être lisible et **relisible**
  (par les autres codeurs ou par nous dans le futur)
- Pour prendre dés le début de **bonnes pratiques**
  (on s'entraîne sur des fonctions simples, mais les problèmes du monde réel sont plus complexes et nécessitent des explications)
- Pour être intégré dans une **bibliothèque** (= collection de fonctions prêtes à être utilisées par des programmes)
- Parce que ça permet de **décrire** le problème
  (et donc de le résoudre plus directement)

Comment ?

Source :
https://lesjoiesducode.fr/?s=commentaire

Sans commentaire

Les Joies du Code, commit du 18 Nov 2021

Quand je commente mon code

Exercice 1 : IMC

affichage textuel : non demandé

```
27  # Fonction de saisie
28  def saisir_imc():
29      print("Entrer le poids en kg :")
30      masse = float(input())
31      print("Entrer la taille en mètres :")
32      taille = float(input())
33      resultat = imc(masse, taille)
34      print("L'IMC est de classe", resultat)
35  #saisir_imc()
```

Fonctions

“ Des questions ? ”