Chapitre 27

Algorithmique et fonctions

I. Notion de fonction

On peut simplifier l'écriture des programmes en utilisant des fonctions, sur le modèle des fonctions numériques étudiées en mathématiques.
Vous avez déjà rencontré diverses fonctions prédéfinies comme `print()`, `input()` ou `range()`.
Lorsqu'une tâche doit être réalisée plusieurs fois par un programme avec seulement des paramètres différents, on peut l'isoler au sein d'une fonction.

Définition 1:

Une fonction est un bloc d'instructions qui a reçu un nom, dont le fonctionnement dépend d'un certain nombre de paramètres (les arguments de la fonction) et qui renvoie le plus souvent un résultat (au moyen de l'instruction retourne).

Exemple 1:

On peut définir la fonction « somtrois » qui, à partir de la donnée de trois nombres a, b et c (les paramètres), renvoie comme résultat la somme de ces nombres.
Ainsi, **retourne**(somtrois(4,2,1)) renvoie la valeur 7.

Fonction somtrois(a, b, c)
  $y = a + b + c$
  retourne(y)

Syntaxe : La syntaxe Python pour la définition d'une fonction est la suivante :

```
def nom_fonction(liste de parametres):
        bloc d_instructions
```

Vous pouvez choisir n'importe quel nom à l'exception des mots-clés réservés du langage.
Comme les instructions `if`, `for` et `while`, l'instruction `def` se termine obligatoirement par un deux-points **:**, qui introduisent un bloc d'instructions qui est précisé grâce à l'indentation.

Exercice 1 :

```
def som(n):
        s=0
        for k in range(1,n+1):
                s=s+k
        return(s)
```

1. On donne le programme ci-contre :
2. Recopier ce programme sur EduPython. Saisissez dans la console python `>>> som(10)`. Qu'affiche le programme ?
3. Quel problème permet de résoudre cet algorithme ?
4. Comment peut-on avec cette fonction calculer la somme des entiers naturels compris entre 50 et 200 ?

Exercice 2 :

On donne le programme de la fonction sous, écrite en langage Python, où n désigne un entier naturel non nul.

```
def reste(n,d):
        r=n%d
        return(r)

def sous(n):
        d=n-1
        while reste(n,d)!=0:
                d=d-1
        return(d)
```

Quel est le rôle de la fonction sous ?

Exercice 3 :

1. Proposer le programme d'une fonction nommée mini, écrit en langage Python, qui retourne le plus petit de deux nombres donnés.
2. Proposer le programme d'une fonction qui retourne le plus petit de quatre nombres donnés, en faisant appel à la fonction mini.

II. Quelques algorithmes du programme

Exercice 1 : Tester si un nombre est premier

1. Rappeler ce qu'est un nombre premier.
2. Donner un algorithme simple permettant de déterminer si un entier naturel est premier.
3. Le programme suivant sera recopier et compléter au fur et à mesure.

```
1 def premier(nombre):
2     ntest=...
3     reponse=True
4     while ... :
5         if ... :
6             reponse=False
7         ntest=ntest+1
8     return(...)
```

   Un **booléen** est un type de données qui ne peut prendre que deux valeurs : vrai ou faux.
4. À l'aide de l'instruction a % b qui donne le reste de la division euclidienne d'un nombre entier a par un nombre entier b, compléter le test de la ligne 5.
5. Quelle valeur doit-on affecter à la variable ntest à la ligne 2 ?
6. Compléter la ligne 4 du programme.
7. Tester la primalité de 7 ; 29 ; 91 ; 529.

Exercice 2 : Recherche de racine de deux par balayage

Le nombre $\sqrt{2}$ est irrationnel : on ne peut pas l'écrire sous forme de fraction.
On cherche à déterminer, par balayage, un encadrement de $\sqrt{2}$ d'amplitude inférieure ou égale à 10^{-n}.

1. Donner en encadrement par deux entiers de $\sqrt{2}$ d'amplitude 1.
2. On souhaite déterminer l'encadrement souhaité par balayage. Une méthode consiste,à partir d'une valeur bien choisie, à tester les carrés de toutes les valeurs avec un pas de 10^{-n}.

   a. Le programme suivant sera recopier et compléter au fur et à mesure.

```
1 def encadrement(n):
2     x=...
3     pas=...
4     while ... :
5         x=...
6     print(...,"< racine carree de 2 <",...)
```

   b. Quelle valeur faut-il affecter à la variable x pour l'initialiser à la ligne 2 ?

   c. Compléter par le pas souhaité à la ligne 3.

   d. Compléter le critère d'arrêt de la boucle tant que à la ligne 4.

   e. Compléter les lignes 5 et 6 pour obtenir le résultat recherché.

   f. À l'aide de la fonction encadrement, déterminer un encadrement de $\sqrt{2}$ à 10^{-5}.

Exercice 3 : Déterminer la première puissance d'un nombre positif donné supérieure ou inférieure à une valeur donnée

1. L'algorithme suivant permet de déterminer la plus petite puissance de 2 supérieure ou égale à 10 000.

1	**Variables :**	n un entier
2	**Initialisation :**	Affecter à n la valeur 0
3	**Traitement :**	**Tant que** $2^n < 10\,000$
4		Affecter à n la valeur $n+1$
5		**Fin Tant que**
6	**Sortie :**	Afficher n

   a. Programmer cet algorithme.

   b. Quelle sera la dernière valeur de n calculée par l'algorithme?

2. Modifier l'algorithme précédent pour obtenir la plus grande puissance de 2 inférieure ou égale à 50 000.

Exercice 4 : Étudier l'alignement de trois points dans le plan

Dans cet exercice, vous allez importer le module math. Il s'agit en fait d'une sorte de bibliothèque qui est un ensemble de fonctions. L'une des grandes forces de Python est le nombre important de bibliothèques disponibles.

On peut citer le module random qui permet d'utiliser des fonctions générant des nombres aléatoires, le module turtle qui permet de réaliser des dessins géométriques, le module numpy qui permet de faire du calcul scientifique, le module sympy qui permet de faire du calcul formel et le module matplotlib qui permet de faire des graphiques.

Pour importer le module math, on peut écrire : from math import*.

L'étoile * indique que l'on souhaite importer toutes les fonctions du module.

Ce module permet d'avoir accès aux fonctions mathématiques et en particulier à la fonction racine carrée (sqrt sur python). Par exemple, sqrt(2) signifie $\sqrt{2}$.

Si on ne voulait importer que la fonction sqrt, on écrirait : from math import sqrt.

1. Le programme à compléter ci-dessous doit permettre d'afficher les coordonnées ainsi que la norme d'un vecteur. Il ne faut pas se fier au nombre de lignes actuel.

```
1 from math import*
2 def vecteur(xA,yA,xB,yB):
3     ...
4     return x,y,n
```

   Compléter ce programme sachant que les variables x, y et n affichées en sortie correspondent respectivement aux coordonnées et à la norme du vecteur.

2. Rédiger et tester un programme permettant de vérifier si deux vecteurs connus par les coordonnées de leurs extrémités sont colinéaires ou non. Dans le traitement des données de ce programme, on utilisera une instruction conditionnelle Si ... Alors ... Sinon.

3. Adapter et tester le programme précédent pour vérifier l'alignement de trois points.

Exercice 5 : Déterminer une équation de droite passant par deux points donnés.

1. Compléter le programme suivant dont l'algorithme permet d'afficher l'équation d'une droite passant par les points $A\left(x_A\,;\,y_A\right)$ et $B\left(x_B\,;\,y_B\right)$.

```
1 def droite(xA,yA,xB,yB):
2     a=...
3     b=...
4     print('y=',a,'x+',b)
```

2. **a.** Tester le programme avec les points $A(2\,;\,-1)$ et $B(3\,;\,5)$.

   b. Expliquer pourquoi le programme retourne une erreur avec les points $A(3\,;\,-2)$ et $B(3\,;\,2)$.

3. Corriger le programme afin qu'il tienne compte de la situation rencontrée dans la question 2b.

Exercice 6 : Pour une fonction dont le tableau de variations est donné, approximation numérique d'un extremum

On se donne une fonction f définie sur un intervalle $[a\,;\,b]$ et son tableau de variations. L'objectif est de créer un algorithme permettant de déterminer une valeur approchée d'un extremum de la fonction f sur l'intervalle $[a\,;\,b]$.

Une méthode consiste à subdiviser l'intervalle $[a\,;\,b]$ en n intervalles de même longueur $\dfrac{b-a}{n}$. On fera ensuite le balayage des valeurs prises par la fonction en chacune des bornes de la subdivision.

L'algorithme ci-dessous, écrit en langage naturel, traduit cette méthode.

1	**Variables :**	a, b, n, min, max, p, x et y sont des réels
2	**Entrée :**	Saisir a, b et n
3	**Initialisation :**	Affecter à min la valeur $f(a)$
4		Affecter à max la valeur $f(a)$
5		Affecter à p la valeur $\dfrac{b-a}{N}$
6		Affecter à x la valeur a
7	**Traitement :**	**Pour** i **allant de** 1 **à** n
8		Affecter à x la valeur $x+p$
9		Affecter à y la valeur $f(x)$
10		**Si** $y > max$
11		**alors** affecter à max la valeur y
12		**Si** $y < min$
13		**alors** affecter à min la valeur y
14		**Fin Si**
15		**Fin Pour**
16	**Sortie :**	Afficher min et max

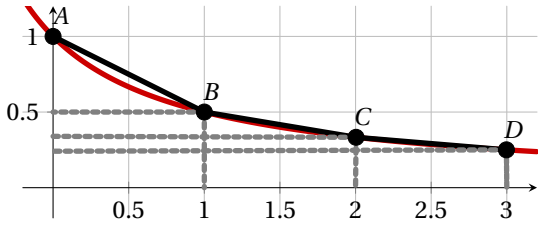
À l'aide d'une calculatrice ou d'un logiciel, écrire et tester un programme traduisant cet algorithme pour la fonction f définie sur l'intervalle $[0\,;\,3]$ par $f(x) = x^3 - 3x^2 + 2x + 5$. On pourra choisir différentes valeurs de N pour affiner le pas. On donne :

x	0	$1-\dfrac{\sqrt{3}}{3}$	$1+\dfrac{\sqrt{3}}{3}$	3
f	0 ↗	$f\left(1-\dfrac{\sqrt{3}}{3}\right)$ ↘	$f\left(1+\dfrac{\sqrt{3}}{3}\right)$ ↗	11

Exercice 7 : Calcul approché de longueur d'une portion de courbe représentative de fonction

Dans un repère orthonormé, on veut calculer, sur l'intervalle $[0\,;\,3]$, une valeur approchée de la longueur de la courbe de la fonction f définie par $f(x) = \dfrac{1}{x+1}$.

1. Pour cela, on a placé sur la courbe quatre points A, B, C et D d'abscisses respectives 0, 1, 2 et 3 formant trois segments $[AB]$, $[BC]$ et $[CD]$. En calculant la somme $AB + BC + CD$ donner une première approximation de la longueur de la courbe de la fonction f sur l'intervalle $[0\,;\,3]$.



2. Une meilleure approximation s'obtient avec un plus grand nombre de points sur la courbe dont les abscisses sont réparties régulièrement sur l'intervalle $[0\,;\,3]$. L'algorithme à compléter suivant permet d'obtenir une approximation de la longueur de la courbe de la fonction f sur l'intervalle $[0\,;\,3]$ en fonction du nombre n de segments ainsi formés.

a. i. Que permet de calculer la variable p.

ii. Compléter la ligne 10 de l'algorithme.

b. Programmer et tester l'algorithme précédent pour différentes valeurs de n.

c. Adapter le programme pour obtenir une approximation de la longueur de la courbe de la fonction f sur l'intervalle $[1\,;\,5]$. Donner cette approximation.

1	**Variables :**	l, p, n, $x1$, $x2$, $y1$ et $y2$ sont des réels
2	**Entrée :**	Saisir n
3	**Initialisation :**	Affecter à l la valeur 0
4		Affecter à p la valeur $\dfrac{3}{n}$
5		Affecter à $x1$ la valeur 0
6		Affecter à $x2$ la valeur $x1 + p$
7	**Traitement :**	**Pour** i **allant de** 1 **à** n
8		Affecter à $y1$ la valeur $f(x1)$
9		Affecter à $y2$ la valeur $f(x2)$
10		Affecter à l la valeur $l + \ldots$
11		Affecter à $x1$ la valeur $x1 + p$
12		Affecter à $x2$ la valeur $x2 + p$
13		**Fin Pour**
14	**Sortie :**	Afficher l