Flot d'exécution

1 Programmation procédurale

Fonction

Exercice 1 : Convertir l'heure

Écrivez une fonction `conversion_heure(n)` prenant en entrée un entier donnant le nombre de secondes qui s'est écoulé depuis minuit et renvoyant un `tuple` donnant l'heure au format heure, minute, seconde. Par exemple `conversion_heure(4567)` devra renvoyer le tuple `(1, 16, 7)`.

Liste

Ordre d'évaluation

2 Programmation structurée

Branchement

Exercice 2 : Trier 3 éléments

Écrire une fonction `tri(a, b, c)` qui prend en argument trois nombres réels a, b et c et qui renvoie le triplet formé de cet 3 éléments, triés par ordre croissant.

Exercice 3 : Chevauchement

Écrire une fonction `chevauche(a, b, c, d)` prenant en entrée quatre entiers a, b, c et d et renvoyant `True` si les segments d'extremités a et b d'une part et c et d d'autre part ont une intersection non vide.

Boucle for

Exercice 4 : Graphisme en console

1. Définir une fonction `triangle1(n)` qui prend en argument un entier n et qui dessine dans le shell un triangle sur n lignes

```
In [1]: triangle1(5)
*
**
***
****
*****
```

2. Définir une fonction `triangle2(n)` qui dessine ce même triangle mais dans l'autre sens.

```
In [2]: triangle2(5)
*****
****
***
**
*
```

3. Définir une fonction `pyramide1(n)` qui dessine une pyramide sur $2n-1$ lignes.

```
In [3]: pyramide1(5)
*
**
***
****
*****
****
```

```
***
**
*
```

4. Définir une fonction `pyramide2(n)` qui dessine une pyramide de la manière suivante.

```
In [4]: pyramide2(5)
    *
   * *
  * * *
 * * * *
* * * * *
```

Exercice 5 : Le Rot13

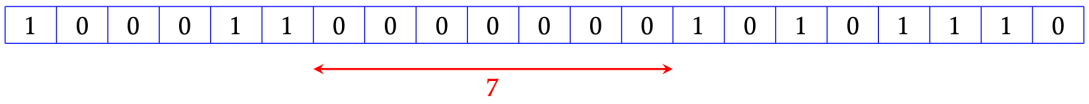
Le Rot13 (rotate by 13 places) est un cas particulier du chiffrage de César. Comme son nom l'indique, il s'agit d'un décalage de 13 caractères de chaque lettre du texte à chiffrer : a devient n, b devient o, c devient p, etc. Son principal aspect pratique est que le codage et le décodage se réalisent exactement de la même manière puisque notre alphabet comporte 26 lettres. Rot13 est parfois utilisé dans les forums en ligne comme un moyen de masquer la réponse à une énigme, un spoiler, ou encore une expression grossière.

1. Écrire une fonction `decale(c)` prenant en entrée un caractère et renvoyant ce même caractère codé en Rot13 si c est un caractère entre `a` et `z` et qui renvoie c sinon. On pourra utiliser les fonctions `ord` et `chr`.
2. Définir une fonctin `rot13(s)` prenant en entrée une chaine de caractères et renvoyant cette même chaine de caractères codée en Rot13.
3. Utilisez cette fonction pour connaitre la réponse à l'énigme suivante : Quelle est la différence entre un informaticien et une personne normale ?

`har crefbaar abeznyr crafr dh'ha xvyb-bpgrg rfg étny à 1000 bpgrgf, ha vasbezngvpvra rfg pbainvaph dh'ha xvybzèger rfg étny à 1024 zègerf.`

Exercice 6 : Plus grand plateau

On considère une liste `a` dont les éléments sont égaux aux entiers 0 ou 1. Rédiger une fonction `pg_plateau(a)` calculant le nombre maximal de 0 consécutifs présents dans cette liste. Par exemple, pour la liste suivante, la fonction devra renvoyer la valeur 7.



Réduction

Exercice 7 : Somme

Écrire une fonction calculant la somme de tous les entiers inférieurs ou égaux à n inclus qui sont multiples de 3 ou de 5.

Exercice 8 : Suite

Écrire une fonction permettant de calculer le n-ième terme de la suite définie par

$$\forall n \in \mathbb{N}, \quad u_n \coloneqq \sum_{k=0}^{n} \frac{1}{k!}.$$

Exercice 9 : Moyenne, variance

Dans cet exercice, on souhaite calculer la moyenne et la variance d'une liste de nombres flottants.

1. Écrire une fonction `moyenne(t)` qui renvoie la moyenne de la liste `t`.
2. La variance d'une famille finie $t \coloneqq (t_0, \dots, t_{n-1})$ est donnée par

$$\mathbb{V}(t) \coloneqq \frac{1}{n} \sum_{k=0}^{n-1} \left(t_k - \overline{t}\right)^2$$

où $\overline{t}$ est la moyenne de t.

(a) Écrire une fonction `variance1(t)` calculant la variance de la liste `t`.

Cette formule nécessite deux parcours de la liste : un pour calculer $\bar{t}$, et l'autre pour calculer $\mathbb{V}(t)$. Pour calculer $\mathbb{V}(t)$, on peut aussi utiliser la formule de Koenig-Huygens

$$\mathbb{V}(t) = \left[\frac{1}{n}\sum_{k=0}^{n-1} t_k^2\right] - \bar{t}^2.$$

(b) Prouver cette formule.

(c) S'en servir pour écrire une fonction `variance2(t)` qui n'effectue qu'un seul parcours de la liste.

Exercice 10 : Palindrome

Un *palindrome* est un mot pouvant se lire dans les deux sens comme : radar, rotor, kayak. Écrire une fonction `palindrome(c)` qui prend en entrée une chaine de caractères `c` et qui renvoie le booléen `True` si cette chaine est un palindrome et `False` sinon. On rappelle que si `s` est une chaine de caractères, on accède au caractère d'indice k grâce à `s[k]`.

Exercice 11 : Monotonie

1. Écrire une fonction `est_croissante(a)` prenant en entrée une liste d'entiers `a` et renvoyant `True` si cette liste est triée dans l'ordre croissant et `False` sinon.
2. Écrire une fonction `est_monotone(a)` prenant en entrée une liste d'entiers `a` et renvoyant `True` si cette liste est triée dans l'ordre croissant ou décroissant et `False` sinon.
3. Écrivez une fonction répondant à la question précédente mais ne parcourant qu'une seule fois la liste.

Boucle while

Exercice 12 : Constante d'Euler

On note pour tout $n \in \mathbb{N}^*$

$$S_n \coloneqq \sum_{k=1}^{n} \frac{1}{k}, \qquad u_n \coloneqq S_n - \ln(n) \quad \text{et} \quad v_n \coloneqq u_n - \frac{1}{n}.$$

On admet que (u_n) et (v_n) tendent vers la même limite γ appelée constante d'Euler et que l'on a

$$\forall n \in \mathbb{N}^*, \quad v_n \leqslant \gamma \leqslant u_n.$$

Écrire une fonction qui calcule un encadrement de γ de largeur inférieure à ε. Notre fonction renverra un tuple des deux réels encadrant γ.

Exercice 13 : Nombre univers

On appelle *nombre univers* (en base 10) un nombre réel dont la partie décimale contient n'importe quelle succession de chiffres de longueur finie. Un exemple simple de nombre univers en base 10 est la constante de Champernowme $0.12345678910111213141516171819202 1\ldots$ On pense que π est un nombre univers mais personne n'a pour le moment réussi à le démontrer. De même, on appelle *suite univers* (en base 10) une suite de nombres entiers elle que n'importe quelle succession de chiffres de longueur finie se trouve dans l'un des termes de cette suite. Il a été prouvé que la suite des puissances de 2 est une suite univers.

1. Écrire une fonction `univers(s)` prenant en entrée une chaine de caractères ne comportant que des chiffres et renvoyant la plus petite valeur de n pour laquelle `s` est présent dans l'écriture décimale de 2^n.
2. Déterminer la plus petite puissance de 2 contenant votre date de naissance au format JJMMAAAA.

Exercice 14 : Nombres premiers

Le but de cet exercice est de déterminer les 1000 plus petits nombres premiers. Pour déterminer si un nombre est premier, on utilise le critère suivant : un entier p est premier si et seulement si $p \geqslant 2$ et lorsqu'il n'est divisible par aucun entier k tel que $2 \leqslant k \leqslant p$.

1. Écrire une fonction `premier(p)` prenant en paramètre un entier p et qui renvoie le booléen `True` lorsque p est premier et le booléen `False` dans le cas contraire.
2. En remarquant qu'il suffit de montrer que p n'admet aucun diviseur k tel que $2 \leqslant k$ et $k^2 \leqslant p$ pour montrer que p est premier, écrire une nouvelle fonction `premier_bis(p)` plus efficace.
3. Utiliser cette fonction pour afficher les mille plus petits nombres premiers.

4. La conjecture de Goldbach postule que tout entier pair supérieur à 3 peut s'écrire comme somme de deux nombres premiers (éventuellement égaux). Vérifier cette conjecture pour tout entier inférieur où égal à 1000.
5. À contrario, montrer que la conjecture suivante est fausse : tout nombre impair est la somme d'une puissance de 2 et d'un nombre premier.

Exercice 15 : Suite de Conway

Les premiers termes de la suite de Conway sont $1, 11, 21, 1211, 111221, \ldots$ chaque terme étant obtenu en lisant à haute voix le terme précédent. C'est pourquoi Conway avait baptisé cette suite *look and say*. Par exemple, le terme 1211 se lit « un 1, un 2, deux 1 » donc le terme suivant est 111221.

1. Écrire une fonction `lookandsay(s)` prenant en paramètre une chaine de caractères représentant un entier et renvoyant la chaine de caractère représentant l'entier suivant dans la suite de Conway.
2. À l'aide de cette fonction, afficher les 20 premiers termes de la suite de Conway.
3. Il a été démontré que si on note u_n le nombre de chiffres du n-ième nombre de Conway, le rapport u_{n+1}/u_n admet une limite finie l. Donnez une valeur approchée de l.
4. Une autre propriété de cette limite est qu'elle ne dépend pas de la valeur initiale (excepté 22). Le vérifier expérimentalement.
5. Démontrer que dans a suite de Conway, ne peuvent apparaitre que les chiffres 1, 2 et 3.

Boucles imbriquées

Exercice 16 : Doublon

Écrire une fonction `doublon(a)` prenant en entrée une liste `a` et renvoyant `True` si `a` possède un doublon et `False` sinon. Par exemple, `doublon([3, 4, 7, 3, 2])` devra renvoyer `True` car 3 est présent deux fois dans la liste.

Exercice 17 : Somme

Écrire une fonction `somme(a, s)` prenant en valeur une liste d'entiers `a` et un entier `s` et qui renvoie `True` si s est la somme de deux entiers de la liste a et `False` sinon. Par exemple `somme([1, 7, 2, 4], 11)` devra répondre `True` car $7 + 4 = 11$ et `somme([1, 7, 2, 4], 14)` devra répondre `False`.