Premiers Algorithmes

PCSI/PTSI

On utilisera pour ce TP l'éditeur en ligne : Future Coder

Notions travaillées :
- Objets (entiers, booléens)
- Variables
- Fonctions
- Flux d'exécution
- Environnements
- Compréhension, implémentation et production d'algorithmes

I Etude d'un algorithme

On considère l'algorithme écrit en pseudo-code ci dessous :

Algorithme F

ENTRÉES : n un entier naturel

1. $f \leftarrow 1$
2. $i \leftarrow 0$
3. **tant que** $i < n$ **faire**
4. $\quad i \leftarrow i + 1$
5. $\quad f \leftarrow f * i$
6. **fin tant que**
7. **renvoyer** f

Exercice 1. *Simulation Papier*

On simule l'algorithme F sur l'entrée 5.

1. Combien y a-t-il de tours de boucles ? (nombre de fois que l'on exécute le block de la boucle).
2. Quelles sont les valeurs de i et f avant chaque exécution de la ligne 3 ?
3. Donnez une équation vérifiée par i et f avant chaque exécution de la ligne 3 ?

Exercice 2. *Spécification de* F

Décrire la sortie de F sur une entrée quelconque n, entier naturel.

Exercice 3. *Vérification*

Implémenter l'algorithme sur `Future Coder` et faire un appel sur l'entrée 5. A l'aide du bouton `snoop` d'abord, puis du bouton `Python Tutor`, vérifiez les résultats de votre simulation.

II Booléens

Exercice 4. *Observation*

Dans la console, observer la valeur de retour des expressions suivantes :

```
2 < 3
2 == 3
1 < 2 and 2 < 3
1 < 2 or 2 == 3
1 < 2 and 2 == 3 or 1 == 1
True or False
True and True
True and False or False and True
y = 2 < 3
x = 2 == 3
x, y
x and y
```

Exercice 5. *Fonction*

Implémenter une fonction `est_diviseur` qui prend en entrée deux entiers naturels a et b et qui renvoie un booléen vrai ssi a est un diviseur de b. (et qui renvoi donc faux si a n'est pas un diviseur de b, quoi d'autre ?)

Exercice 6. *Premier*

Un nombre entier naturel plus grand que 2 est premier ssi il ne possède aucun diviseurs parmis les entiers naturels strictement plus grand que 1 *et plus petits ou égaux à sa racine carrée.*

Complétez et implémentez l'algorithme suivant pour qu'il renvoie un booléen vrai ssi n est premier.

Vous *devrez* utiliser la fonction `est_diviseur`.

Algorithme P

ENTRÉES : n un entier naturel plus grand ou égal à 2

```
p ← …
d ← 2
tant que … faire
    si … alors
        p ← …
    fin si
    d ← d + 1
fin tant que
renvoyer p
```

Simulez l'algorithme sur `Python Tutor` sur l'entrée 11. Combien d'environnements sont créés au total ? Combien y en a-t-il au plus existants simultanément ?

III Premiers algorithmes

Exercice 7. *Méthode de Halley pour le calcul de* $\sqrt{2}$

Implémenter en Python la fonction `halley` qui calcule le $n^{\text{ème}}$ terme de la suite récurrente suivante :

$$H_0 = 1\ ,\ H_{n+1} = H_n \cdot \frac{(H_n^2 + 6)}{(3\,H_n^2 + 2)}$$

Exercice 8. *Suite de Fibonacci*

Implémentez en Python la fonction `fibonacci` qui calcule le $n^{\text{ème}}$ terme de la suite récurrente d'ordre 2 suivante :

$$F_0 = 0\ ,\ F_1 = 1\ ,\ F_{n+2} = F_{n+1} + F_n$$

Exercice 9. *Syracuse*

Les mathématiques ne sont pas encore prêtes pour de tels problèmes – Paul Erdõs

Une suite de Syracuse est définie par la relation de récurrence suivante :

$$S_{n+1} = \begin{cases} \dfrac{S_n}{2} & \text{si } S_n \text{ est pair,} \\ 3S_n + 1 & \text{si } S_n \text{ est impair.} \end{cases}$$

La conjecture de Syracuse stipule que, quelque soit $S_0 \in \mathbb{N}^*$, la suite de Syracuse débutant par S_0 atteint 1. On notera qu'une fois la valeur 1 atteinte, la suite boucle sur les valeurs $1, 4, 2, 1, \ldots$.

Implémentez une fonction `syracuse` qui prend en entrée un entier strictement positif s_0 et qui renvoie le premier rang auquel la suite atteint la valeur 1.

Est-ce que la fonction `syracuse` termine sur toutes les entrées ?

Exercice 10. *Euclide*

Implémenter en Python l'algorithme d'Euclide qui prend en entrée deux entiers a et b et qui renvoie le `PGCD` de ces deux entiers.

`https://fr.wikipedia.org/wiki/Algorithme_d%27Euclide`

IV Pour s'entraîner

Exercice 11. *Suites adjacentes*

Implémentez en Python la fonction `adjacentes` qui prend en argument un flottant ϵ et qui calcule le premier rang n termes des suites adjacentes suivantes, à partir duquel $|a_n - b_n| < \epsilon$:

$$b_{n+1} = \frac{a_n + b_n}{2}\ ,\ a_{n+1} = \frac{2}{b_{n+1}}\ ,\ a_0 = 1, b_0 = 2$$

Exercice 12. *Méthode de Héron*

La suite x^a définie si dessous converge vers $\sqrt{a}$.

$$\forall n \in \mathbb{N} \quad x_{n+1}^a = \frac{x_n^a + \frac{a}{x_n^a}}{2}\ ,\ x_0^a = a$$

Implémentez une fonction `heron` qui prend en argument deux entiers a et n positifs et renvoie x_n^a.

Trouvez le rang à partir duquel x_n^2 approche $\sqrt{2}$ à 10^{-10} prêt (≈ 1.4142135623).

Exercice 13. *Nombre parfait*

Un nombre parfait est un nombre entier positif égal à la somme de ses diviseurs stricts (lui-même exclu).

Implémentez une fonction `parfait` qui prend en entrée un entier et renvoie vrai s'il est parfait.

V Pour aller plus loin

Exercice 14. *Logarithme entier*

On rappelle que pour tout $n \geq 1$ et $b \geq 2$, le logarithme entier en base b de n est l'unique entier l tel que

$$b^l \leq n < b^{l+1}$$

Implémentez en Python la fonction ayant les spécifications suivantes :

Algorithme `logarithme_entier`

ENTRÉES : $n \geq 1$ et $b \geq 2$ deux entiers
SORTIES/EFFETS : un entier. Le logarithme entier en base b de n

Exercice 15. *Nombres Chanceux d'Euler*

Un nombre chanceux d'Euler est un nombre $n > 1$ tel que pour tout $0 \leq i \leq n-2$, $i^2 + i + n$ est premier.

Implémentez une fonction `chanceux_euler` qui prend en entrée un entier n et renvoie vrai ssi n est chanceux.

Exercice 16. *Fonction 91 de McCarthy*

La fonction 91 de McCarthy est définie pour $n \in \mathbb{N}$ par la formule suivante :

$$f(n) = \begin{cases} n - 10 & \text{si } n > 100 \\ f(f(n + 11)) & \text{sinon.} \end{cases}$$

Implémentez une fonction `mccarthy` qui prend en entrée un nombre entiers n positifs et renvoie $f(n)$. Observer le retour de la fonction pour les valeurs plus petites ou égales à 101.