Fonctions

Exercice 1 – Suite de Syracuse

La suite de Syracuse est une suite d'entiers définie par un premier entier naturel non nul $u_0 = a \in \mathbb{N}^*$ et la relation de récurrence

$$\forall n \in \mathbb{N}\ u_{n+1} = \begin{cases} \frac{u_n}{2} & \text{si } u_n estpair \\ 3 * u_n + 1 & \text{si } u_n estimpair \end{cases}$$

La conjecture de Syracuse affirme que , quelque soit le choix de u_0, il existe un rang p tel que $u_p = 1$. Le premier entier p vérifiant u_p=1 est parfois appelé "longueur de vol".

- Écrire une fonction syracuse qui, étant donné un entier a, renvoie l'entier a/2 si a est pair, et l'entier 3a+1 sinon.
- Écrire une fonction test_syracuse qui teste la conjecture de Syracuse pour un entier naturel donné

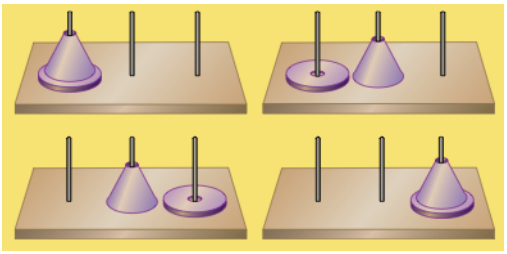
Exercice 2 – Retour sur la tour de Hanoi

On dispose de trois piquets (que l'on nommera D, I , A) et de n pièces de bois circulaires de diamètre respectivement 1, 2, ... n, trouées en leur milieu de telle sorte qu'on puisse « enfiler » chaque pièce de bois sur n'importe lequel des trois piquets.

Au début du jeu, les pièces de bois sont toutes empilées sur le piquet D, par ordre de diamètre croissant (la plus grande en bas de la pile, la plus petite en haut de la pile). Le but est de réussir à déplacer les pièces de bois pour obtenir le même empilement sur le piquet D, en respectant la contrainte suivante : une pièce de bois ne peut pas être empilée sur une autre pièce de bois dont le diamètre serait plus petit.

Autrement dit, à chaque étape du jeu, les pièces empilées sur chaque piquet sont par ordre de diamètre croissant (avec le plus grand en bas).

Voici un algorithme de résolution pour déplacer n disques d'un piquet d'origine à un piquet de destination :



si n = 1 : déplacer l'unique disque (situé sur le piquet d'origine) sur le piquet de destination, et c'est fini.

si n > 1 :

– appliquer l'algorithme de résolution pour déplacer les (n-1) disques du piquet d'origine au piquet intermédiaire (qui n'est ni le piquet d'origine ni celui de destination finale, mais qui devient le nouveau piquet de destination dans cette étape).

– déplacer le n-ième disque sur le piquet de destination ;

– appliquer à nouveau l'algorithme de résolution pour déplacer les (n-1) disques du piquet intermédiaire au piquet destination. Le piquet intermédiaire devient le nouveau piquet origine pour cette étape.

1. Écrivez une fonction récursive hanoi qui prend en argument un entier n, et qui affiche toutes les instructions pour passer de l'état initial (les n pièces empilées sur le piquet A) à l'état final (les n pièces empilées sur le piquet C) en respectant les règles.
2. Calculer le nombre de coups nécessaires (de deux manières différentes)

Exercice 3 – Jeu de N.I.M. (aussi appelé jeu des allumettes)

Il s'agit de réaliser un programme pour jouer au jeu des allumettes ou jeu de Nim . Ce jeu très connu et dont il existe de nombreuses variantes se jouera de la manière suivante :
on dispose d'un certain nombre d'allumettes. Chacun son tour, un des deux joueurs prend 1 , 2 ou 3 allumettes selon son choix. Celui qui est obligé de prendre la dernière allumette perd la partie.
Il est demandé de faire un programme pour jouer à ce jeu avec deux joueurs "humains".

1. Écrire la fonction affiche(n) : cette fonction qui prend en paramètre un entier n doit afficher sur une ligne n étoiles séparées par des espaces puis aller à la ligne.
2. Écrire la fonction min(a,b) qui prend en paramètre deux entiers renvoie le plus petit des deux.
3. Écrire la fonction saisie(a,b) qui prend en paramètre deux entiers , demande à l'utilisateur de saisir un entier entre a et b (compris) et recommence tant que la réponse n'est pas satisfaisante. Si la réponse est satisfaisante, l'entier est renvoyé. On supposera que l'utilisateur saisie effectivement un entier et on ne testera que le fait qu'il soit compris entre a et b.
4. On fera ensuite le programme principal. On jouera avec 21 allumettes au départ. Le programme devra afficher le joueur dont c'est le tour, l'état du jeu, etc.. comme sur l'exemple ci-dessous. On appellera les joueurs joueur1 et joueur2. A tout moment un joueur doit choisir comme nombre d'allumettes un nombre entre 1 et le minimum de nb-1 et de 3 si nb est le nombre d'allumettes restantes.

   Exemple de jeu :

```
* * * * * * * * * * * * * * * * * * * * *
joueur 1 combien d'allumettes, saisir un entier entre 1 et 3: 6
saisir un entier entre 1 et 3:  3

* * * * * * * * * * * * * * * * * * *
joueur 2 combien d'allumettes, saisir un entier entre 1 et 3: 3

* * * * * * * * * * * * * * * * *
joueur 1 combien d'allumettes, saisir un entier entre 1 et 3: 3

* * * * * * * * * * * * *
joueur 2 combien d'allumettes, saisir un entier entre 1 et 3: 3

* * * * * * * * * *
joueur 1 combien d'allumettes, saisir un entier entre 1 et 3 :3

* * * * * *
joueur 2 combien d'allumettes, saisir un entier entre 1 et 3: 3

* * *
```

```
joueur 1 combien d'allumettes, saisir un entier entre 1 et 3: 1

* *
joueur 2 combien d'allumettes, saisir un entier entre 1 et 2: 3

mauvaise saisie , saisir un entier entre 1 et 2: 1

*
victoire du joueur 2
```

5. Bonus : faire une version où un joueur humain joue contre l'ordinateur.

Exercice 4 – Calcul d'un histogramme

Etant donné un tableau `tirages` de taille n(=100 entiers) dont les valeurs sont entre 1 et 6. On souhaite calculer l'histogramme de ce tableau. Pour cela on utilisera un tableau à 6 cases où on stockera le nombre de fois où on a obtenu chacune des 6 faces du dé.

— Ecrire la fonction `histo(tirages, int n)` où n est un entier et tirages un tableau de n valeurs entières (de 1 à 6). La fonction doit renvoyer un tableau h de 6 entiers représentant le nombre de tirage de chaque face (le nombre d'occurrences de chaque chiffre). Le tableau h est calculé par la fonction histo à partir du tableau tirages.

Exercice 5– Nombre d'Armstrong

Définition. On appelle nombre de Armstrong d'un entier naturel (composé de 3 chiffres) un nombre qui est égal à la somme des cubes des chiffres qui le composent.

Par exemple, 153 est un nombre de Armstrong car : $1^3 + 5^3 + 3^3 = 153$.

— Ecrire un programme qui demande à l'utilisateur un entier compris entre 100 et 999 et qui indique si cet entier est un nombre de Armstrong.