---

title: "Fonction Syracuse en Ocaml"
date: "2021-01-05"
categories:

- "informatique"
- "mathematiques"

  tags:

- "OCaml"
  description: ""

---

# Définition de Syracuse

On définit la suite de Syracuse récursivement $\forall n \in \mathbb{N}$ par :

- $u_0 = n$
- $\forall n \in \mathbb{N}^* : u_{n+1} = \begin{cases} \frac{u_n}{2} & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{si } u_n \text{ est impair} \end{cases}$

# Implémentation en OCaml

## La fonction

La fonction Syracuse en elle-même ne comporte aucun piège. Notez que j'utilise un "match" ici. Une conditionnel "if-else" aurait très bien pu faire l'affaire.

```
let rec syracuse n =
    match n with
    1 -> 1
    | n when n mod 2 = 0 -> syracuse(n/2)
    | _ -> syracuse (3*n+1);;
```

Conformément à la conjecture, cette fonction retourne à priori "toujours" la valeur 1 :

```
syracuse 54;;
- : int = 1
```

## Affichage de la suite

Pour afficher la suite, il suffit d'afficher n à chaque appel de la fonctions :

```
let rec printSyracuse n =
    print_int n;
    print_string " ";
    match n with
    1 -> 1
    | n when n mod 2 = 0 -> printSyracuse(n/2)
    | _ -> printSyracuse (3*n+1);;
```

## Calcul du temps de vol

Le temps de vol est définir comme le plus petit indice $n$ tel que $u_n = 1$.

## Calcul du temps de vol en altitude

## Calcul de l'altitude maximale