

# Algorithmique TD1

## L3 Informatique – ENS Cachan

Guillaume Bury

19 septembre 2016

### Exercice 1

On cherche à calculer le PGCD de deux entiers sur une machine rudimentaire (i.e sans utiliser de division). On propose la fonction suivante :

```
function PGCD(x: integer, y: integer) {
  if x >= y then
    return PGCD(y, x)
  else
    return PGCD(x, y - x)
}
```

- Donner la spécification de la fonction de calcul du PGCD
- Corriger la fonction ci-dessus afin qu'elle réponde à la spécification donnée
- Prouver la correction, et la terminaison de la fonction corrigée par rapport à sa spécification

### Exercice 2

On considère la fonction suivante, supposée trouver un entier dans un tableau :

```
function find(t: sorted integer array,
             x: integer, low: integer, high: entier) {
  while low < high do
    tmp <- (low + high) / 2 (* integer division *)
    if t[tmp] < x then
      low <- tmp
    else
      high <- tmp
  done
  return low
}
```

- Donner une spécification de la fonction de recherche d'un entier
- Corriger la fonction ci-dessus afin qu'elle réponde à la spécification donnée
- Prouver la correction, et la terminaison de la fonction corrigée par rapport à sa spécification

### Récréation

Expliquer et prouver le comportement de la fonction 91 de McCarthy :

```
function f91(x: entier) {
  if x > 100 then
    return x - 10
  else
    return f91(f91(x + 11))
}
```

### Exercice 3

On dispose d'un tableau d'entiers  $T[1..n]$ . On cherche à calculer à la fois le minimum et le maximum du tableau.

1. Proposer un premier algorithme. Prouver sa correction. Donner le nombre exact de comparaisons effectuées dans le pire des cas.
2. On peut diminuer le nombre de comparaisons en regroupant les éléments par paires. Proposer un algorithme suivant ce principe et donner le nombre exact de comparaisons effectuées dans le pire des cas.

### Exercice 4

Pour chacun des problèmes suivants, proposer un algorithme itératif et un algorithme récursif, prouver leur correction et leur terminaison, et calculer leur complexité en temps.

1. Calcul du  $n$ -ième terme de la suite de Fibonacci définie par :

$$\begin{cases} u_0 = 0 \\ u_1 = 1 \\ u_n = u_{n-2} + u_{n-1}, \quad \text{pour } n > 1. \end{cases}$$

2. Calcul du PGCD de deux entiers par l'algorithme d'Euclide.

### Exercice 5

1. On considère un tableau  $T[1..n]$  d'entiers naturels. On modifie  $T$  par le procédé itératif suivant : si on peut trouver deux indices  $1 \leq i < j \leq n$  tels que  $T[i] > T[j]$ , alors on échange  $T[i]$  et  $T[j]$  et on essaye à nouveau, sinon on stoppe. Remarquons que ce procédé est non déterministe puisque dans le cas où existent plusieurs paires  $(i, j)$  « telles que... » on peut choisir l'une quelconque de ces paires. Que fait ce procédé ?
2. On modifie la procédure d'échange. Soient deux indices  $i < j$  tels que  $T[i] > T[j]$ . L'échange consiste maintenant à choisir deux nouvelles valeurs  $a_i$  et  $a_j$  telles que  $T[i] \geq a_j \geq a_i \geq T[j]$  et à effectuer  $T[j] \leftarrow a_j$ ,  $T[i] \leftarrow a_i$ . Autrement dit, on peut rapprocher les valeurs des éléments échangés. Par exemple, on pourra passer de  $(0, 30, 20, 40)$  à  $(0, 22, 28, 40)$ . Que peut-on dire de ce nouveau procédé ?
3. On modifie une dernière fois la procédure d'échange. Soient deux indices  $i < j$  tels que  $T[i] > T[j]$ . L'échange consiste maintenant à choisir deux nouvelles valeurs  $a_i$  et  $a_j$  telles que  $T[i] \geq a_j \geq a_i \geq T[j]$  et  $a_j > T[j]$  puis à effectuer  $T[j] \leftarrow a_j$ ,  $T[i] \leftarrow a_i$ . Autrement dit,  $T[j]$  doit forcément croître après l'opération. Que peut-on dire de ce dernier procédé ?