SE2A3 - Programmation avancée

CM2 - Récursivité

Thibault Liétard
Polytech' Lille

2022

Remerciements

Ce cours est largement inspiré du travail de Walter Rudametkin.

Notion de récursivité

Structure de données

Définition
Une structure de données est récursive si elle se définit elle même.

Exemple

```
1 struct cell {
2     int value;
3     struct cell* next;
4 };
5
6 struct list {
7     int nb_val;
8     struct cell* first;
9 };
```

Fonctions récursives

Définition
Une fonction récursive est une fonction qui possède un ou plusieurs cas d'exécutions où elle réalise un appel à elle même avec des paramètres différents. Elle doit nécessairement posséder au moins un cas terminal.

Exemple

Fonction factorielle
$n! = n \times (n-1)!$

Notion de cas terminal

Définition
Un cas terminal (ou cas d'arrêt) est un cas d'exécution de la fonction tel que les paramètres donnés donne une solution immédiate.

Condition
Idéalement, ce cas doit être facilement atteignable à partir d'un cas plus complexe du problème donné.

Exemple

Fonction factorielle

$n! = n \times (n-1)!$

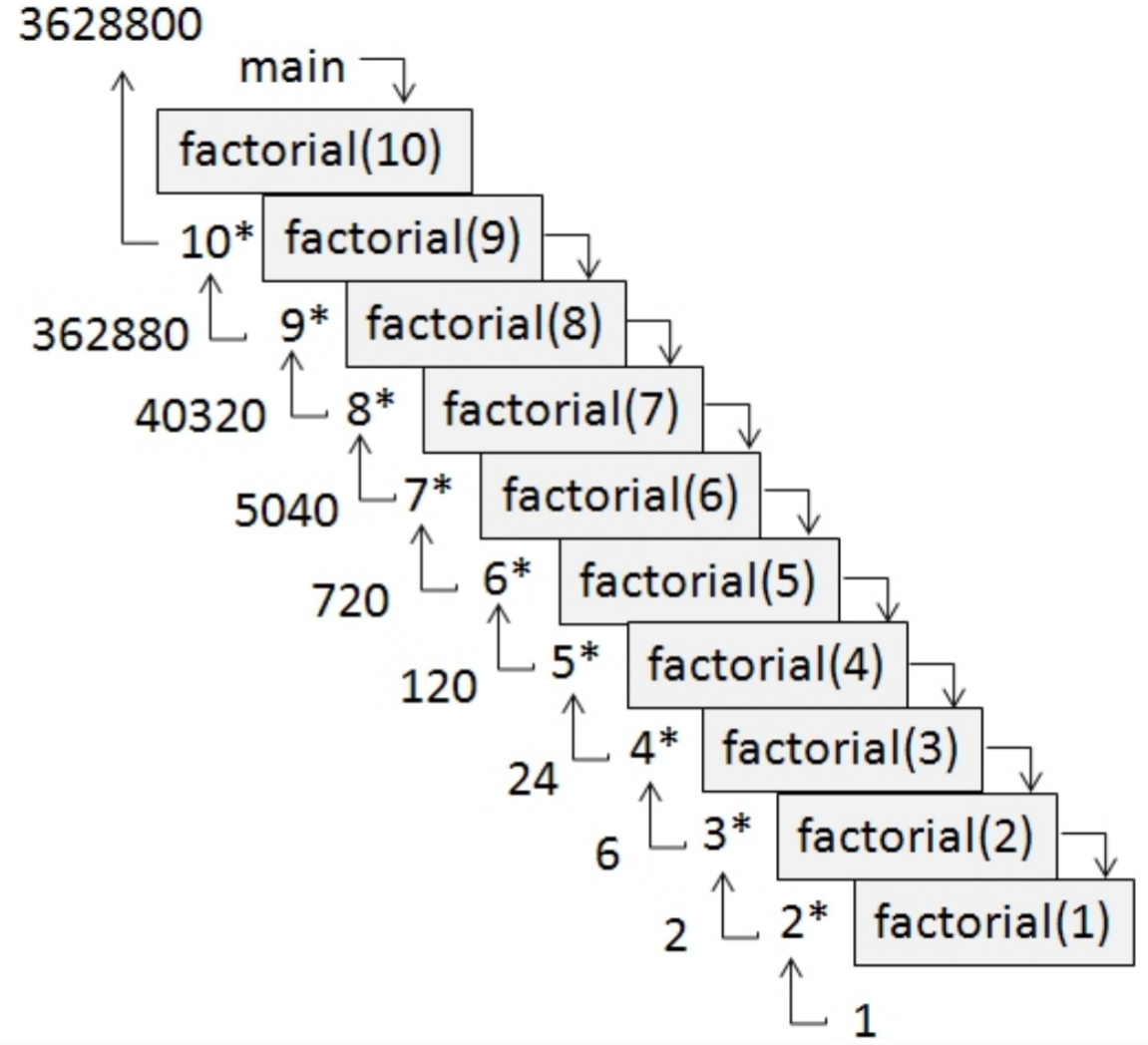
$0! = 1$

Exemple

Algorithme

```
1 int fact(n){
2     assert(n>=0);
3     if (n==0){
4         return 1;
5     }
6     else{
7         return n*fact(n-1);
8     }
9 }
```

Exécution


3628800
main
factorial(10)
10*
factorial(9)
362880
9*
factorial(8)
40320
8*
factorial(7)
5040
7*
factorial(6)
720
6*
factorial(5)
120
5*
factorial(4)
24
4*
factorial(3)
6
3*
factorial(2)
2
2*
factorial(1)
1

Concevoir un algorithme récursif

Distinction

- Cas généraux récursifs
- Cas terminaux non récursifs
- Condition de terminaison

Exemple : recherche dichotomique

Principe
On cherche l'index d'un élément x dans un tableau t d'entiers trié de taille n. On commence par regarder l'élément à l'index n/2. Trois cas sont possible :

- t[n/2] == x : on a trouvé
- t[n/2] < x : on cherche dans le sous tableau de gauche
- t[n/2] > x : on cherche dans le sous tableau de droite

Conception

Distinction des cas

- Cas généraux : t[n/2]>x ou t[n/2]<x
- Cas terminal : t[n/2]==x
- Condition de terminaison : x>t[n-1] (à tester en premier lieu)

Terminaison de fonction

Problème
Dans l'exemple précédent, on suppose que x est nécessairement dans le tableau, ce qui peut ne pas être le cas. Il manque un cas terminal crucial qui peut faire boucler l'algorithme à l'infini.

Conception corrigée

Distinction des cas

- Cas généraux : t[n/2]>x ou t[n/2]<x
- Cas terminal : t[n/2]==x
- Condition de terminaison : x>t[n-1]
- Condition de terminaison : t n’est pas vide

Autres exemples

Fonctions récursives usuelles

- suite de fibonacci
- taille d'une liste
- inversion d'une liste
- parcours d'un arbre
- etc...

This is it ☺