LIFAP6 : Introduction

Vincent Nivoliers

Avant-propos

Interaction :

- posez des questions pendant et après le cours
- je ferai mon possible pour vous y encourager
- fournissez moi un retour sur mon cours
- présence sur le discord de la licence

Avant-propos

Interaction :

- posez des questions pendant et après le cours
- je ferai mon possible pour vous y encourager
- fournissez moi un retour sur mon cours
- présence sur le discord de la licence

Présence en cours :

- les diapos du cours (s'il y en a) ne sont pas exhaustives
- si vous vous organisez pour prendre des notes je peux les relire
- attention vos ordinateurs et téléphones sont distrayants

Intervenants et volume horaire

Cours magistral : 15h

- Vincent Nivoliers

Travaux dirigés : 15h

- Vincent Nivoliers (A)
- Nicolas Louvet (B)
- Lois Paulin (C)

Travaux pratiques : 30h

- Vincent Nivoliers (A1)
- Agathe Herrou (A2)
- Nicolas Louvet (B1)
- Jean-Christophe Mignot (B2)
- Lois Paulin (C1)
- Jean-Claude Iehl (C2)

voir sur Tomuss

Algorithmique, programmation et complexité

Algorithmique

Problème :

- mélanger un tableau

Proposition de solution :

Entrées : tab : un tableau, taille : sa taille
pour *i allant de* 0 *à* taille -1 **faire**
    $i_1 \leftarrow$ nombre aléatoire entre 0 et taille -1
    $i_2 \leftarrow$ nombre aléatoire entre 0 et taille -1
    échanger tab$[i_1]$ et tab$[i_2]$

Algorithmique

Problème :

- mélanger un tableau

Proposition de solution :

Entrées : tab : un tableau, taille : sa taille
pour *i allant de* 0 *à* taille -1 **faire**
    $i_1 \leftarrow$ nombre aléatoire entre 0 et taille -1
    $i_2 \leftarrow$ nombre aléatoire entre 0 et taille -1
    échanger tab$[i_1]$ et tab$[i_2]$

Cette solution est-elle valide ?

Algorithmique

Problème :

- mélanger un tableau ← trop vague !

Proposition de solution :

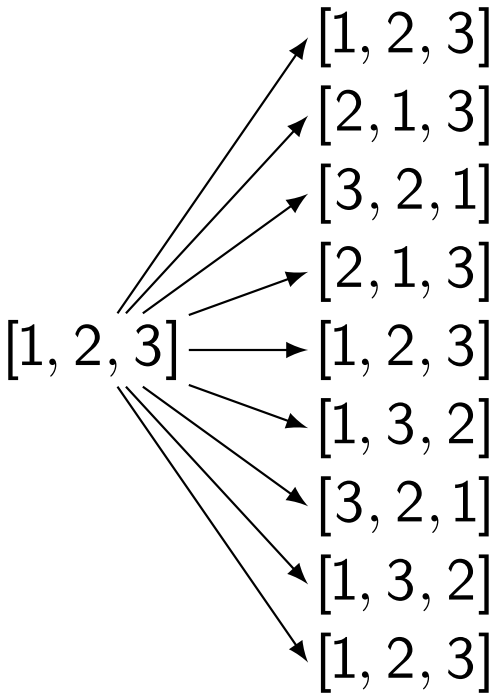
Entrées : tab : un tableau, taille : sa taille
pour *i allant de* 0 *à* taille $-$ 1 **faire**
    $i_1 \leftarrow$ nombre aléatoire entre 0 et taille $-$ 1
    $i_2 \leftarrow$ nombre aléatoire entre 0 et taille $-$ 1
    échanger tab[i_1] et tab[i_2]

Cette solution est-elle valide ?

Algorithmique

Problème :

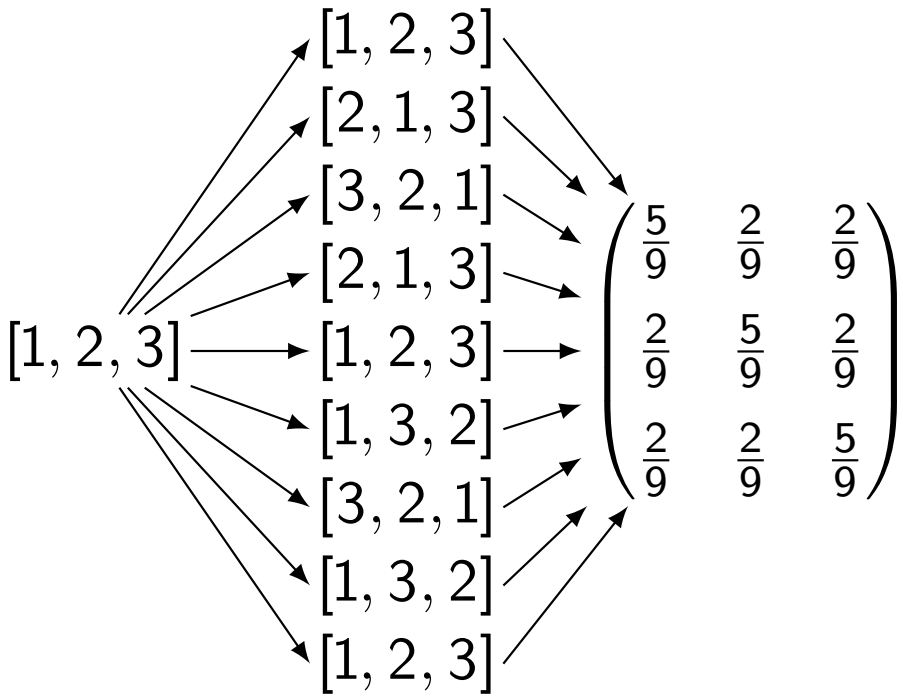
- mélanger un tableau ← trop vague !



Algorithmique

Problème :

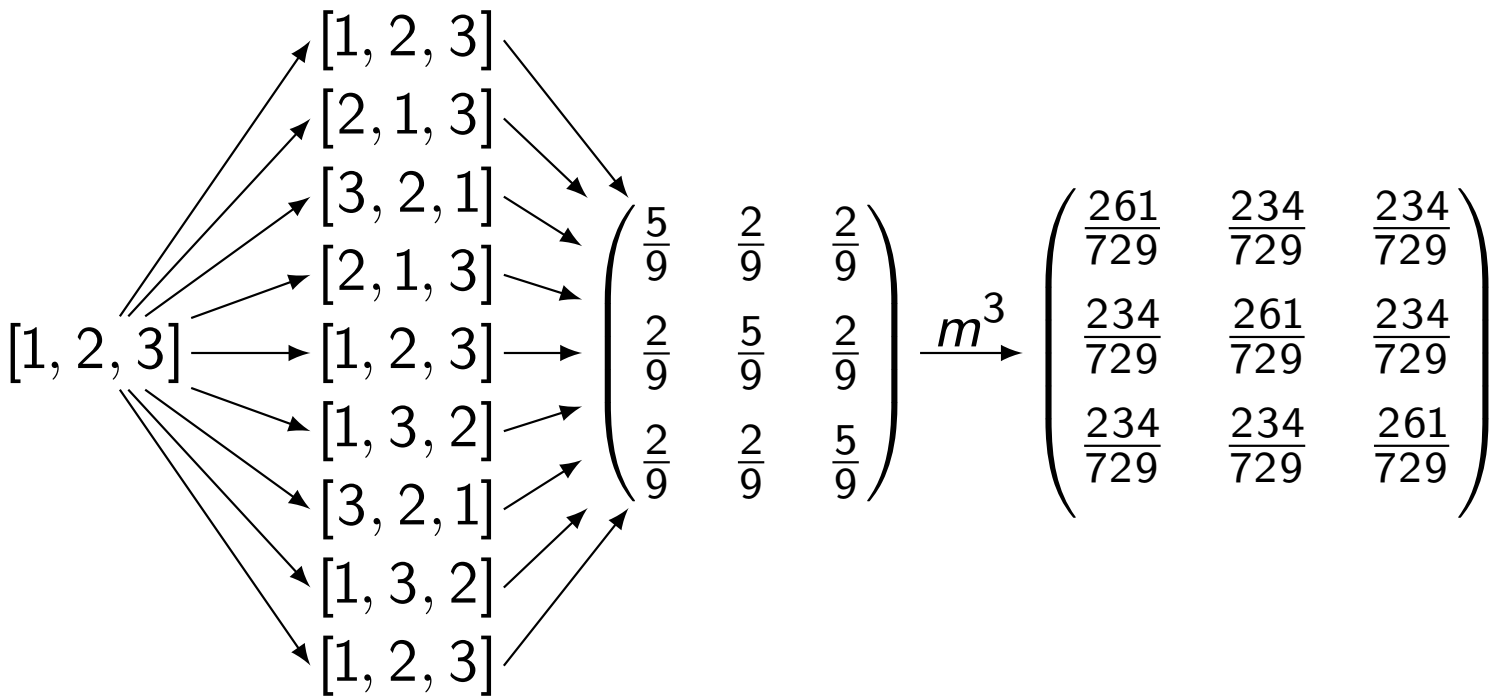
- mélanger un tableau ← trop vague !



Algorithmique

Problème :

- mélanger un tableau ← trop vague !



Toutes les solutions n'ont pas la même probabilité !

Algorithmique

Problème :

- mélanger un tableau
- toutes les permutations ont la même probabilité

Algorithmique

Problème :

- mélanger un tableau
- toutes les permutations ont la même probabilité

Une autre solution :

Entrées : tab : un tableau, taille : sa taille
pour *i allant de* 0 *à* taille $-$ 1 **faire**
    $j \leftarrow$ nombre aléatoire entre 0 et taille $- i - 1$
    échanger tab[j] et tab[taille $- i - 1$]

Cette solution remplit le contrat !

Compétences algorithmiques visées

Prérequis :

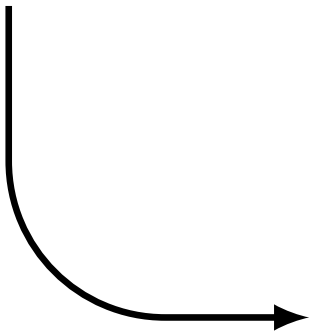
- Conteneurs de base : listes, tableaux, tableaux dynamiques

À la fin de ce cours, vous devriez être capables de :

- Formuler un problème posé avec un vocabulaire précis
- Proposer des solutions pour résoudre des problèmes simples
- Prouver qu'une solution (simple) résoud un problème (simple)

Programmation

Entrées : tab : un tableau, taille : sa taille
pour *i allant de* 0 *à* taille $-$ 1 **faire**
    $j \leftarrow$ nombre aléatoire entre 0 et taille $-$ i $-$ 1
    échanger tab[j] et tab[taille $-$ i $-$ 1]



```
void melange(int* tab, unsigned int taille)
{
  for(unsigned int i = 0; i < taille - 1, ++i) {
    unsigned int index = rand() % (taille - i) ;
    int tmp = tab[taille - i - 1] ;
    tab[taille - i - 1] = tab[index] ;
    tab[index] = tmp ;
  }
}
```

C/C++

Compétences en programmation visées

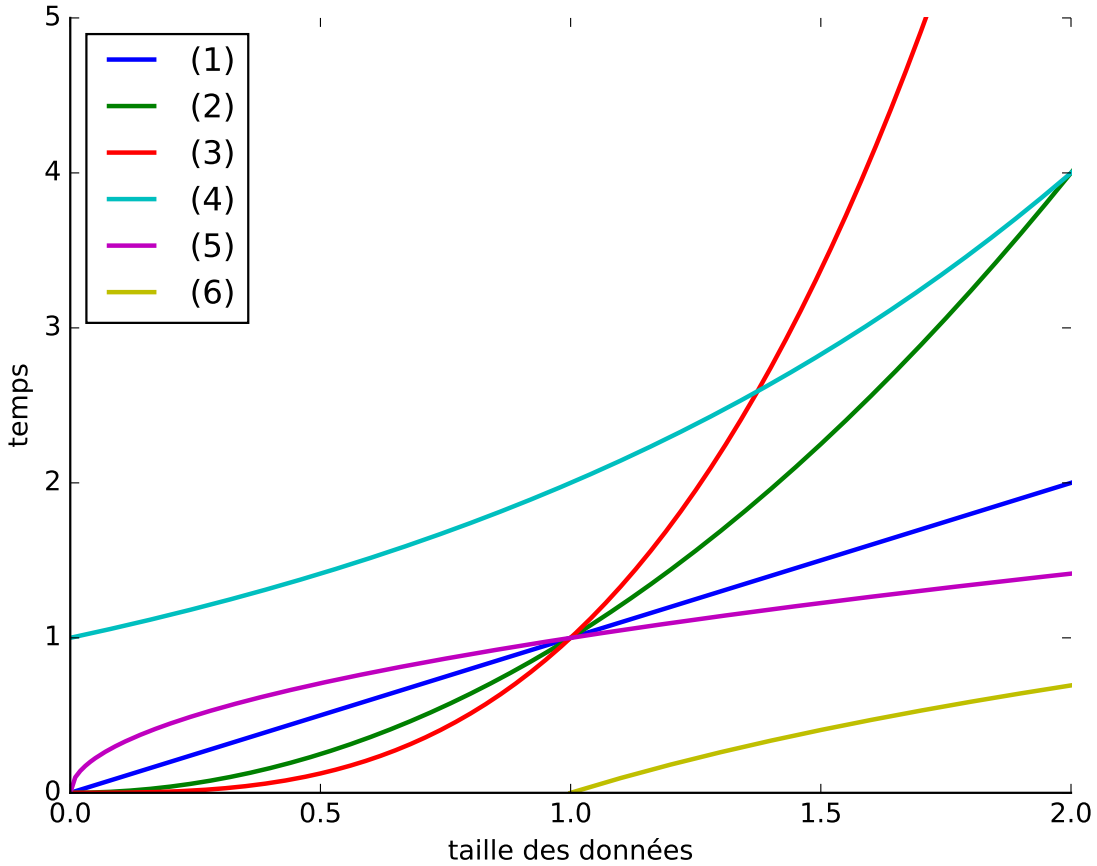
Prérequis :

- Connaissance basique du C++, types primitifs, tableaux, pointeurs
- Maîtrise de votre éditeur de code
- Utiliser des outils de debug : GDB, Valgrind
- Produire et modifier un Makefile pour gérer vos compilations

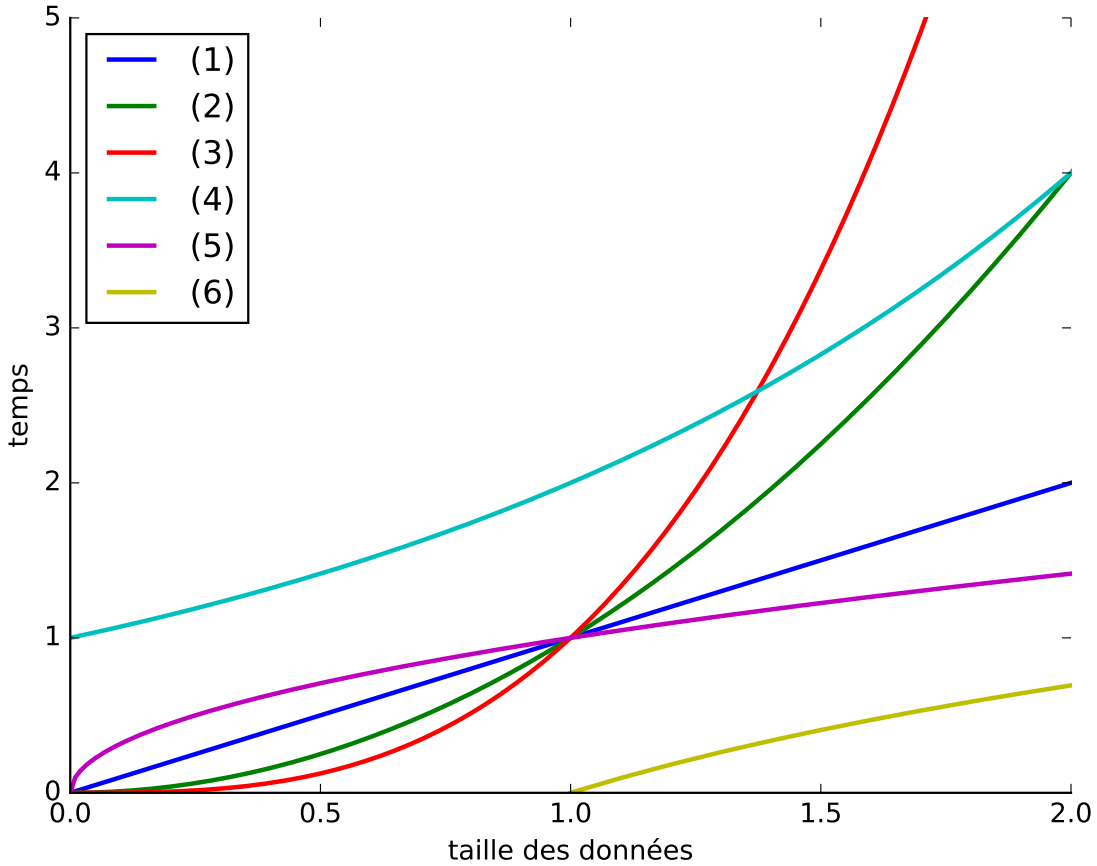
À la fin de ce cours, vous devriez être capables de :

- Construire et parcourir des structures de données complexes
- Gérer proprement la mémoire allouée à votre programme
- Organiser votre code pour faciliter sa réutilisation

Complexité : étude du coût d'un programme

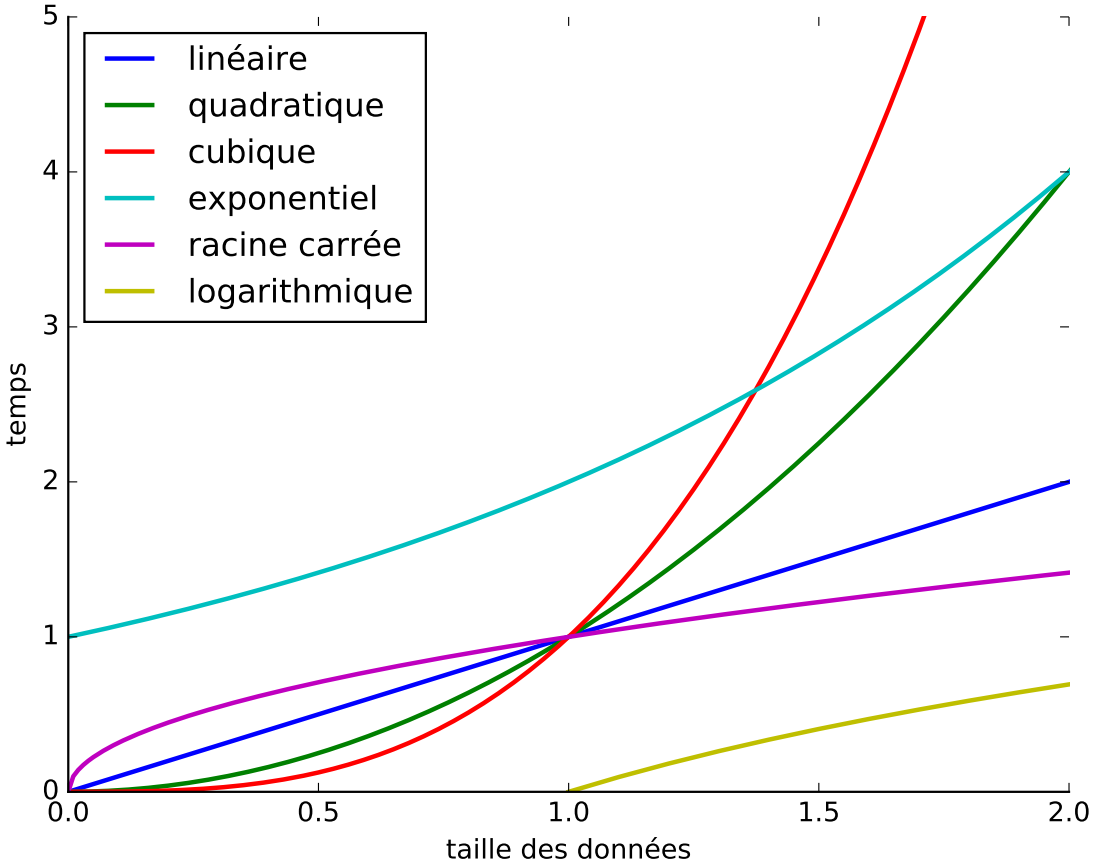


Complexité : étude du coût d'un programme

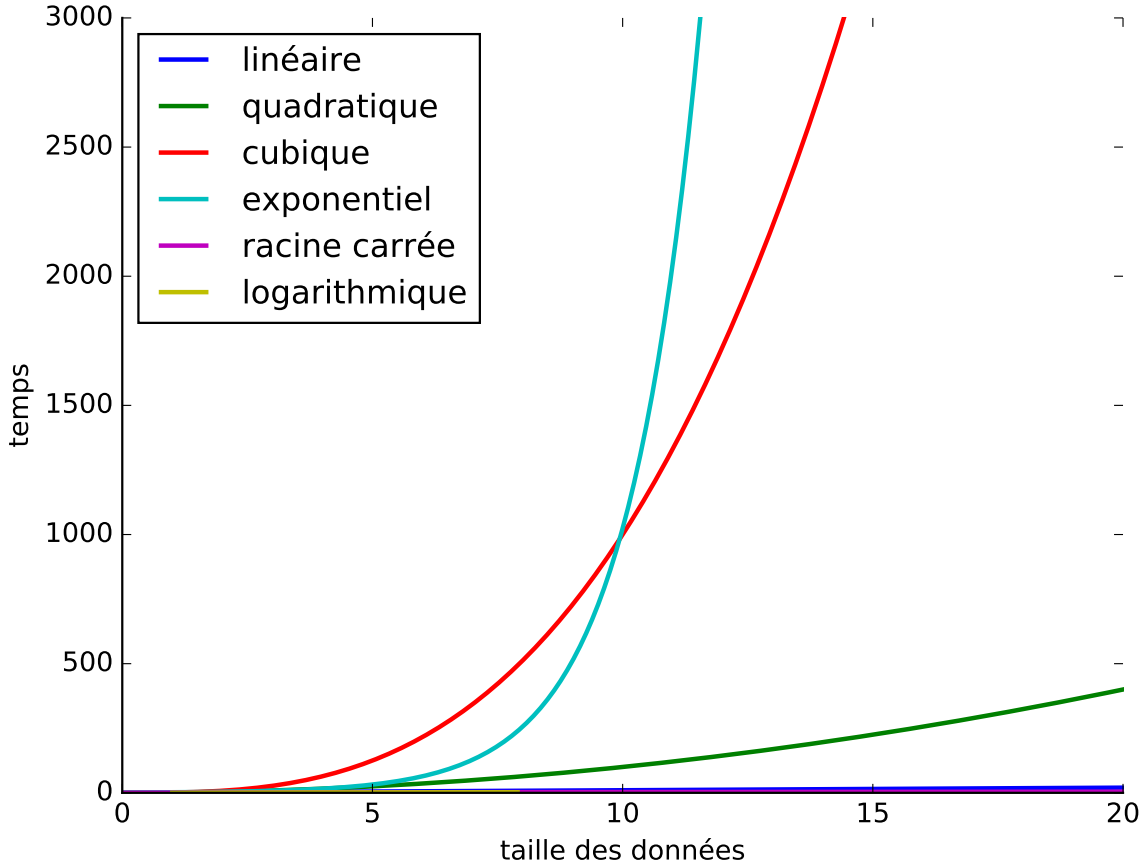


linéaire, quadratique, cubique, logarithmique, exponentiel, racine carrée ?

Complexité : étude du coût d'un programme



Complexité : étude du coût d'un programme



Compétences visées en complexité

Prérequis :

- Formalisme mathématique : somme, pour tout, il existe
- Démonstration par récurrence
- Suites : arithmétiques, géométriques, somme des termes

À la fin de ce cours, vous devriez être capables de :

- Poser des questions pertinentes sur le coût de vos programmes
- Comprendre les structures de données des bibliothèques utilisées
- Reconnaître les classes de complexité classiques

Modalités de contrôle

Contrôle final

- coefficient : 0.3
- épreuve sur table de 2h
- un recto-verso manuscrit A4 autorisé

Interros de TD

- coefficient : 0.2
- 3 épreuves de 20 minutes en début de TD, 2 retenues
- pas de documents autorisés

TP rendu

- coefficient : 0.1
- épreuve de 3h sur une séance complète, rendue en fin de séance
- individuel, pas de communications autorisées

Projet

- coefficient : 0.4
- sujet fourni en début de module, à rendre en fin de module
- séances de TP dédiées à l'avancement du projet
- en binôme

Projet : Magic Maze

conception graphique Gyom

un jeu de Kasper Lapp édité (actuellement) par Sit Down