TAD et Implémentation

Introduction

Manipulation de TAD

- Pour cela, on a besoin seulement :
  - d'une **notation** pour décrire les données
  - des **primitives**
  - de la **sémantique** des primitives

- Écriture alors de divers algorithmes sur les TAD (parcours, recherche, mise à jour, etc…) à l'aide des primitives

Implémentation de TAD

- C’est la détermination :
  - de la représentation concrète des données
  - des algorithmes utilisés au sein des primitives

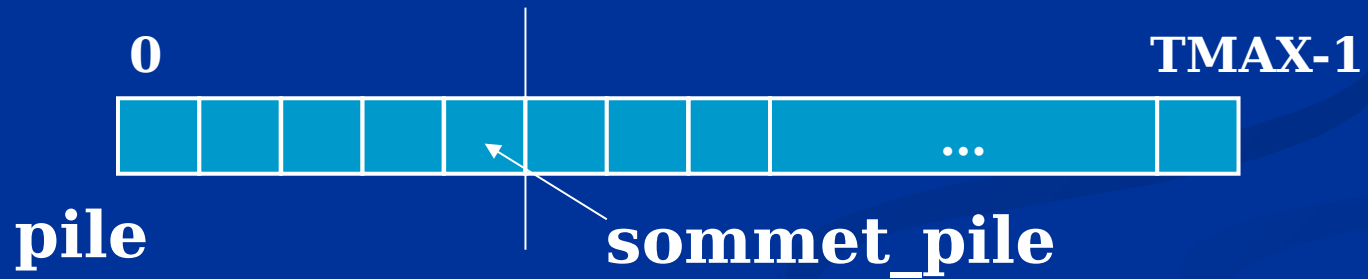
- Contrainte importante : respecter les spécifications abstraites sans négliger l’efficacité
  - s’appuyer sur des types existants adéquats

maitriser les complexités en espace et en temps des algorithmes mis en œuvre

Exemple
le TAD Pile

Sol 1 : allocation statique

- **Tableau de taille variable**, avec une taille maximum « suffisamment » grande…



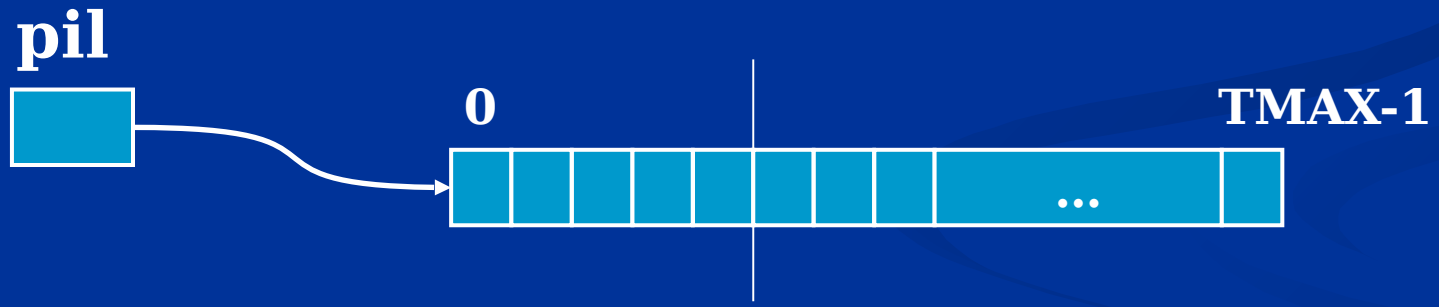
⇒ **limitation** imposée à la taille de la pile !

⇒ rajout d'une primitive spéciale

Fonction **pilePleine**(P : TPile) : booléen

Sol 2 : allocation dynamique

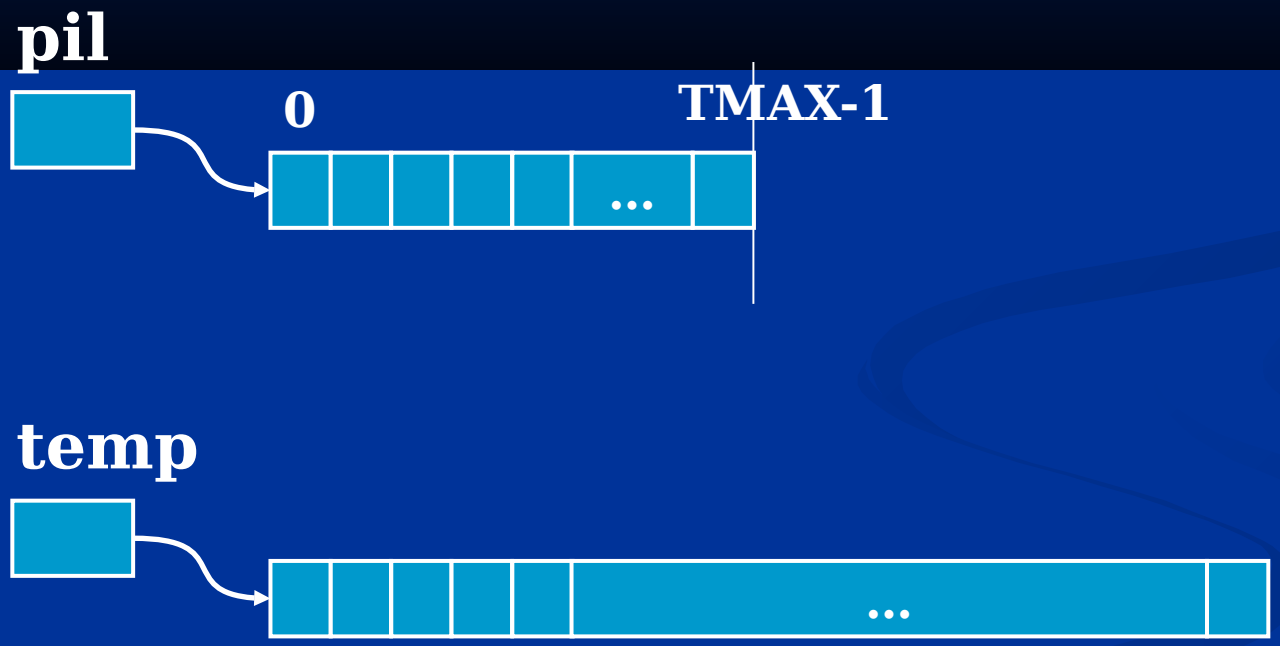
- Le tableau est alloué dynamiquement à la construction de la pile, et désalloué à la destruction



- Lorsque la pile est saturée (≡ tableau plein)
  ⇒ « agrandir » le tableau…
  ⇒ invisible pour l'utilisateur du TAD Pile…

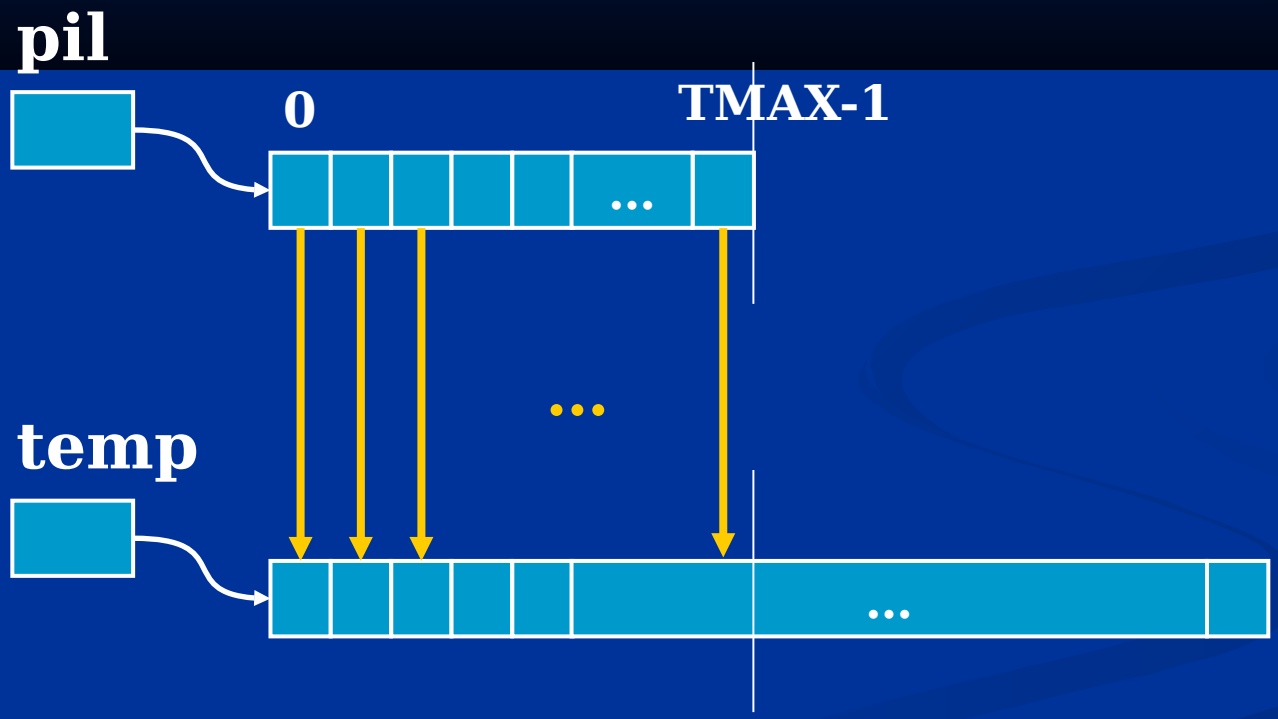
Lorsque la pile est saturée (≡ tableau plein)

- **1.** on alloue un bloc plus grand



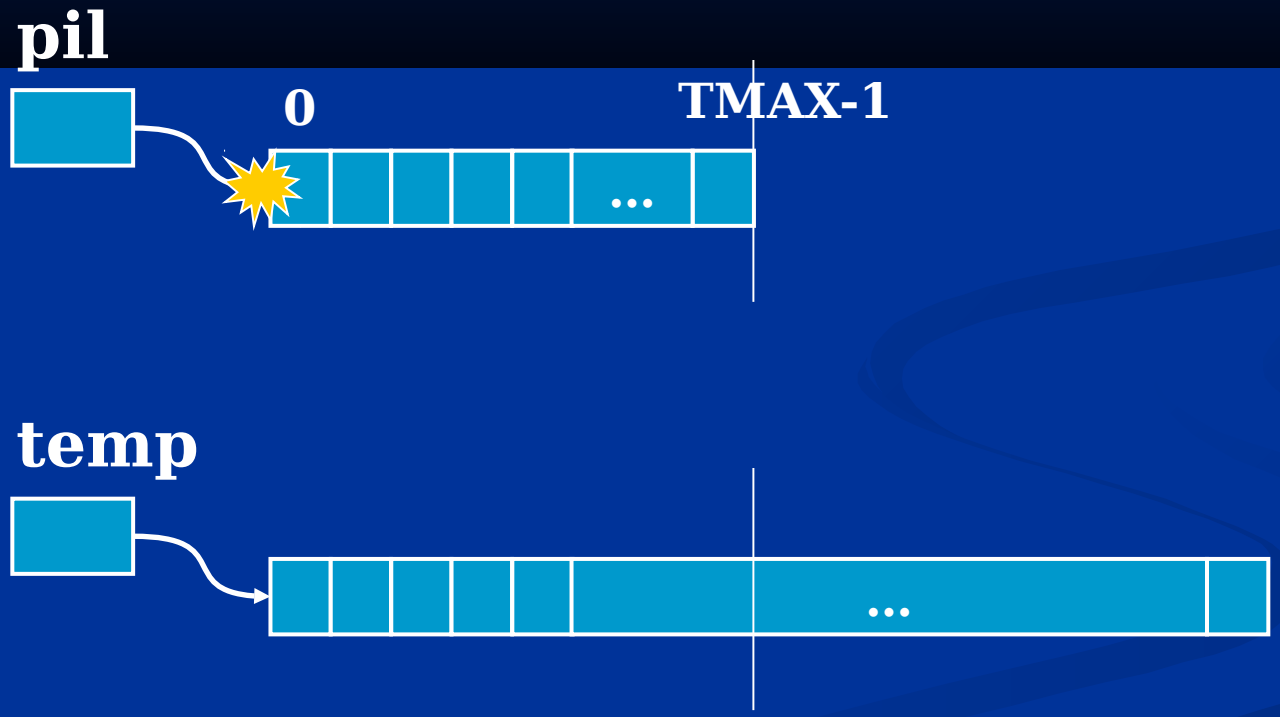
Lorsque la pile est saturée (≡ tableau plein)

- **2.** on y recopie l'ancienne pile



Lorsque la pile est saturée (≡ tableau plein)

- **3.** on libère l’ancien bloc



Lorsque la pile est saturée (≡ tableau plein)

- **4.** on fait pointer la pile sur le nouveau bloc