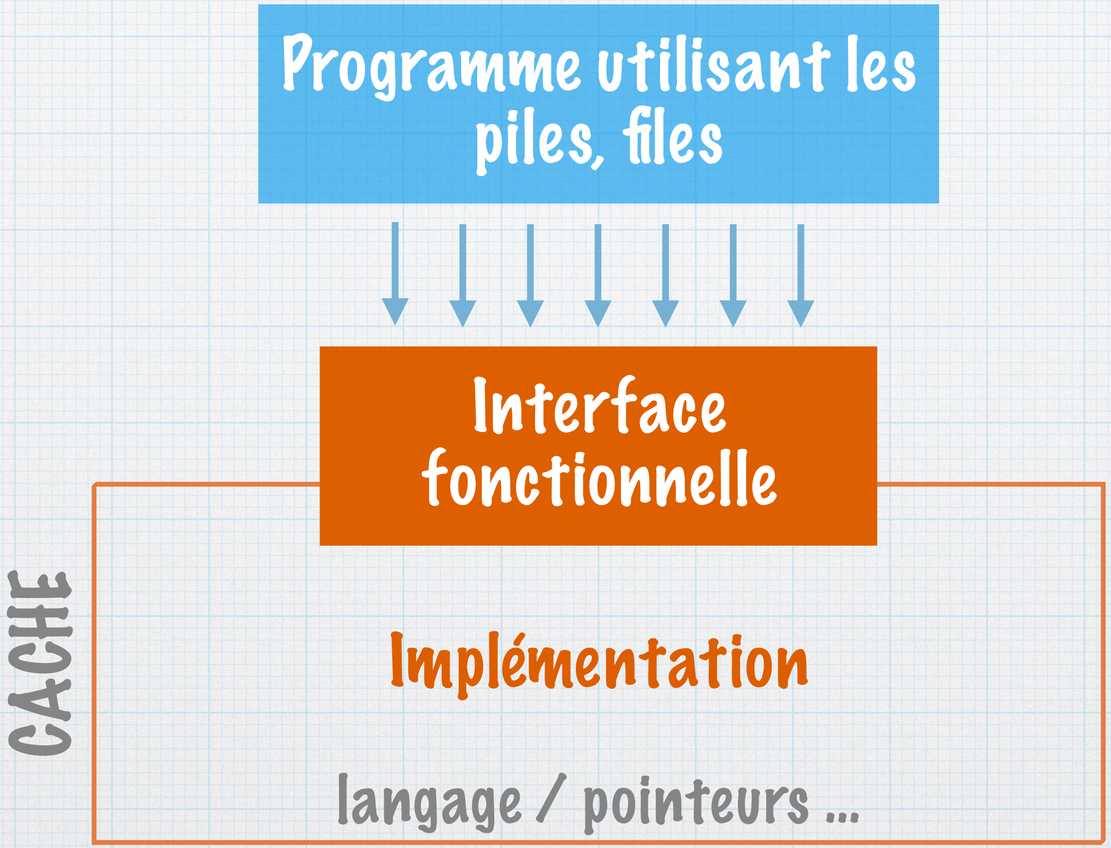
Algorithmique et structures de données

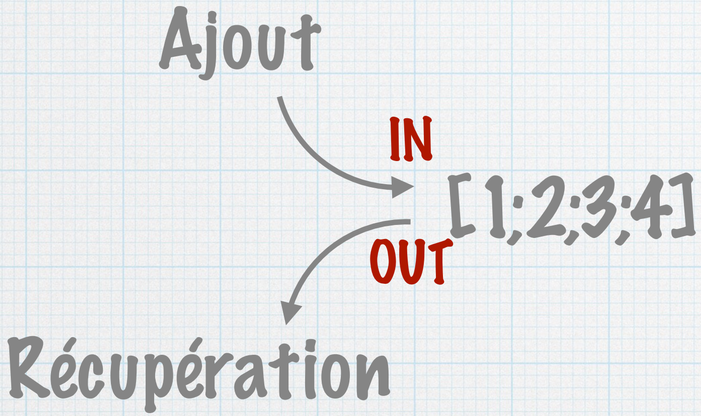
Cours 3 : piles, files

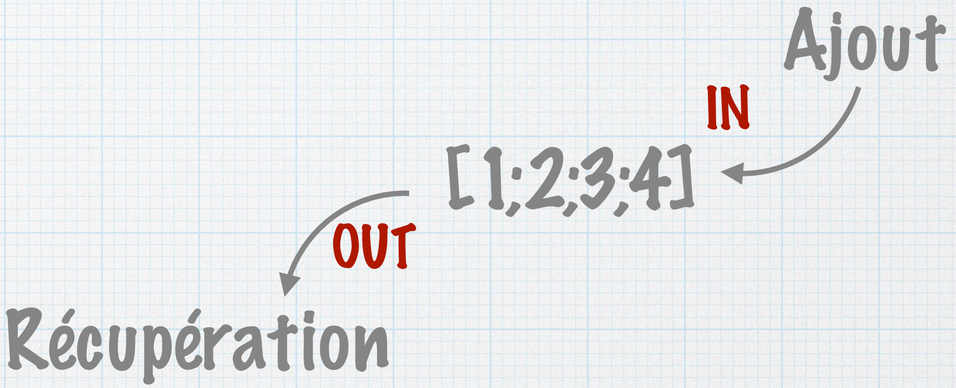
Structures de données : spécification et implémentation



Piles et files (intuition)

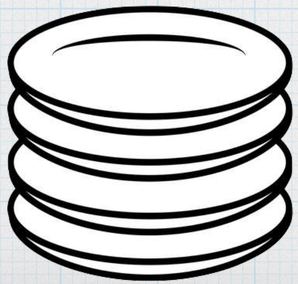
Données stockées séquentiellement





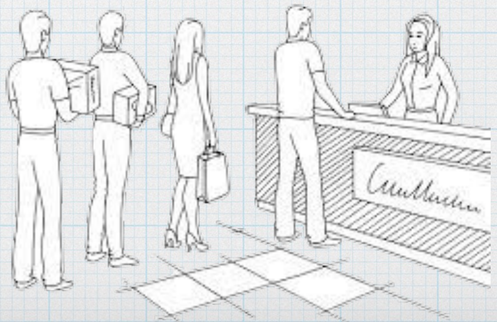
Piles

LIFO : Last In First Out



Files

FIFO : First In First Out



TAD / interface fonctionnelle

Piles

TAD (version fonctionnelle)

```
Pile creer_pile_vide ()
bool est_pile_vide (Pile)
Pile push (Data, Pile)
Data top (Pile)
Pile pop (Pile)
```

Privilégiée pour piles/files

Interface fonctionnelle (version « effet de bord »)

```
Pile creer_pile_vide ()
bool est_pile_vide (Pile)
void push (Data, Pile)
Data top (Pile)
void pop (Pile)
```

```
Stack s ;
s = create_empty_stack() ;

push(1,s) ;
push(2,s) ;

pop(s) ;
```

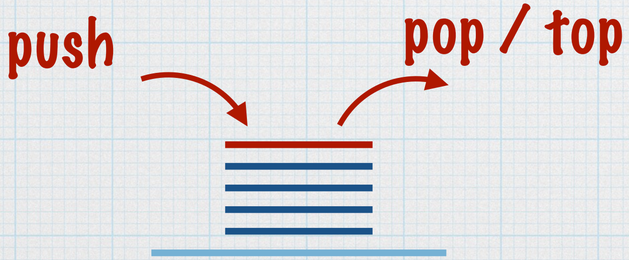
Interfaces fonctionnelles
Piles / Files

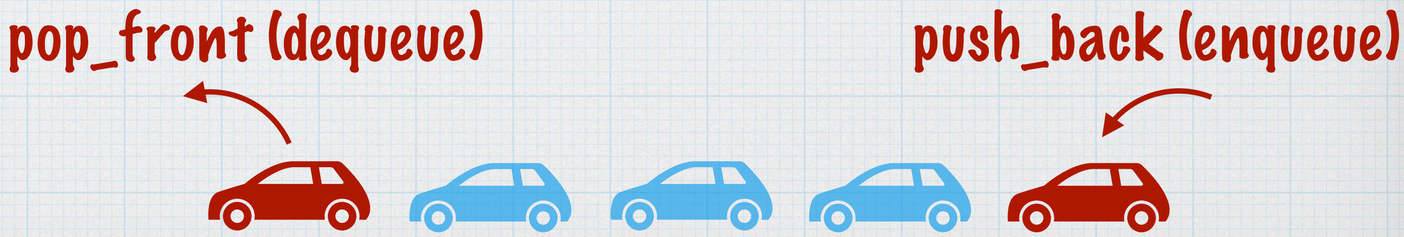
Piles

Stack create_empty_stack ()
bool is_empty_stack (Stack)
void push (Data, Stack)
Data top (Stack)
void pop (Stack)

Files

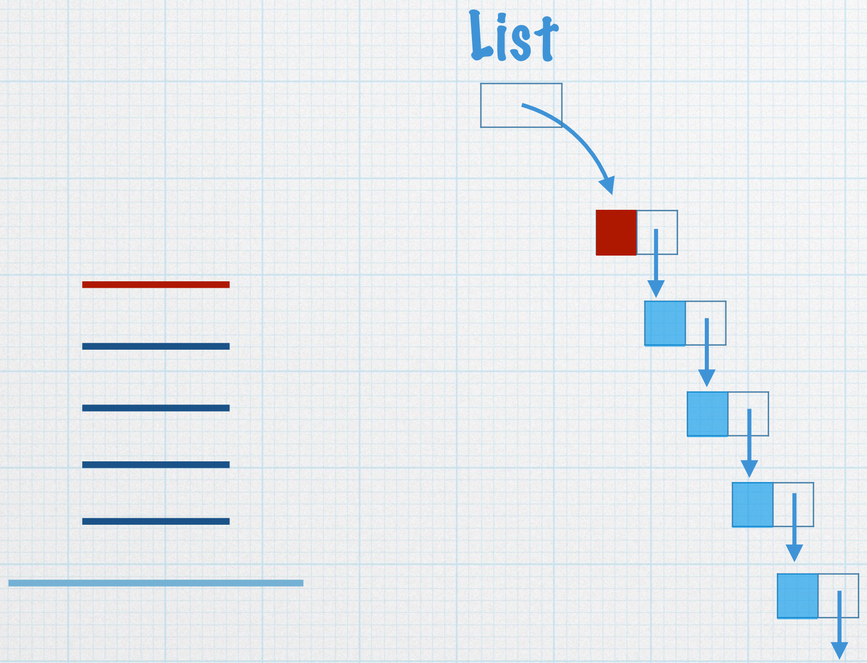
Queue create_empty_queue ()
bool is_empty_queue (Queue)
void push_back (Data, Queue)
Data top (Queue)
void pop_front (Queue)



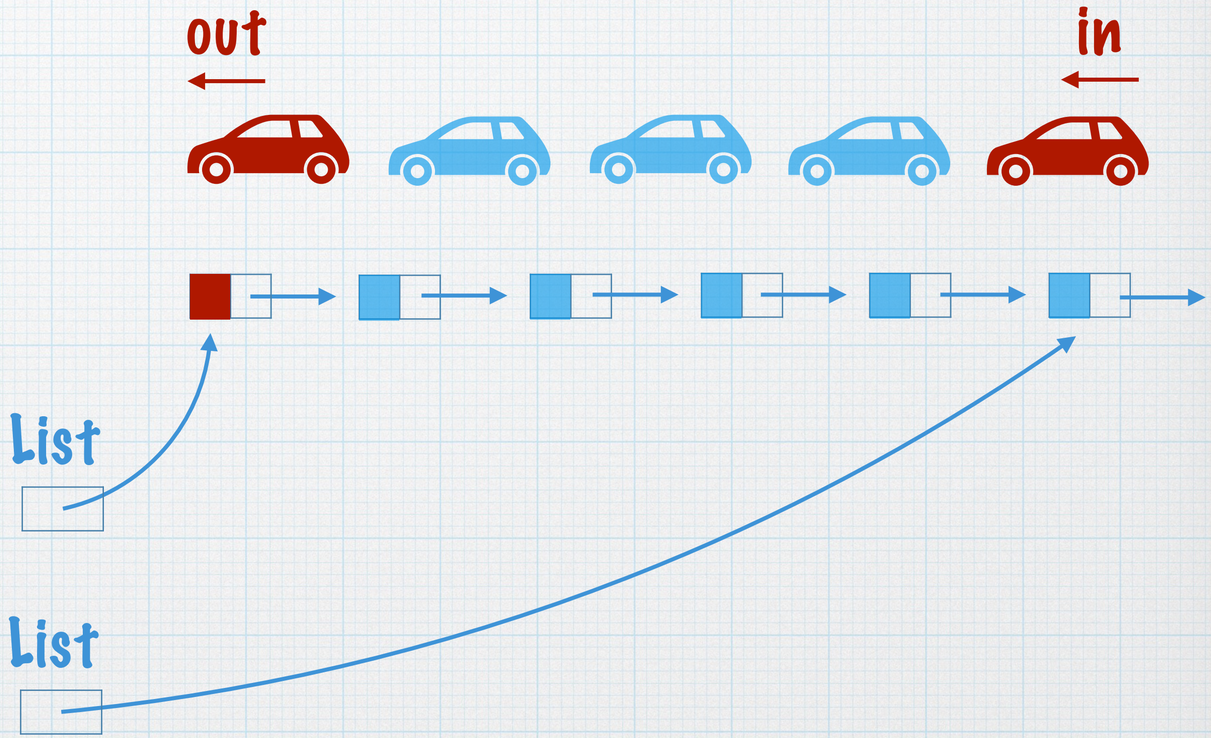


Implémentation

Implémentation des Piles/Files (via Listes)

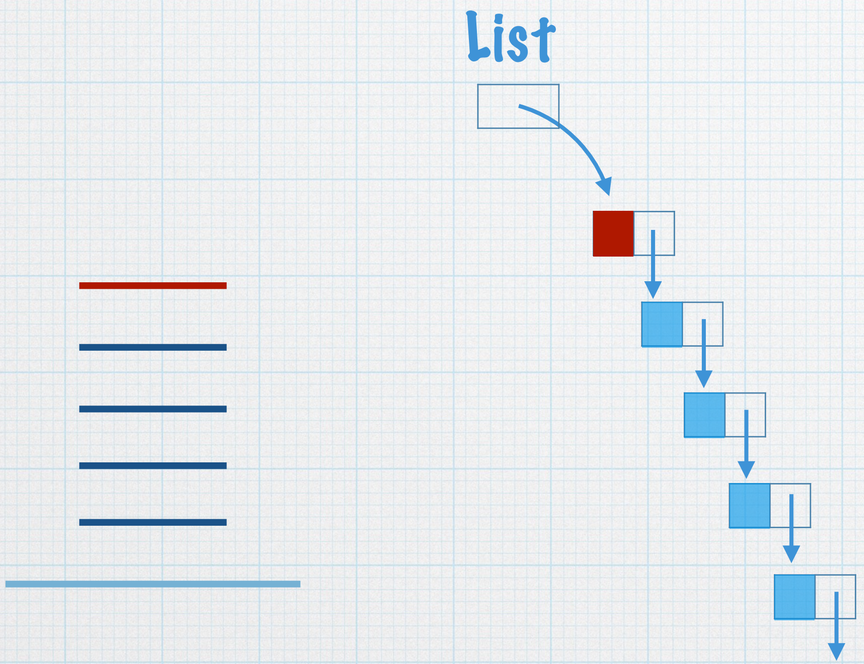


Pile ↔ Liste



File ↔ Liste + ptr sur le dernier élt

Structures de données pour l'implémentation à effet de bord



Pile ↔ Liste

v.0
typedef List Stack ;

Fonctions à effet de bord :
void push (Data, Stack *)
Data top (Stack *)

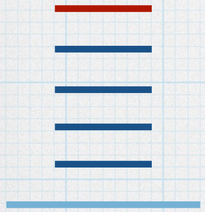
Exemple :
Stack p1, p2 ;
p1 = create_empty_stack() ;
p2 = p1 ;
push(3,p1) ;

p2 est toujours vide …

v.1
struct zstack {
    List top ; } ;
typedef struct stack * Stack ;

Fonctions à effet de bord :
void push (Data, Stack)
Data top (Stack)

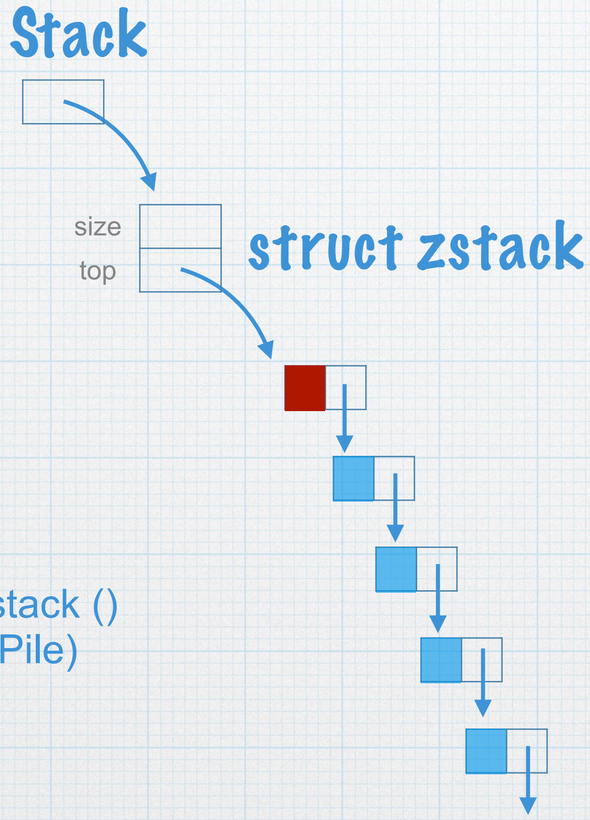
Implémentation des Piles/Files (via Listes)



Pile

```
struct zstack {
    int size ;
    List top ; } ;
typedef struct stack * Stack ;
```

Stack create_empty_stack ()
bool is_empty_stack (Pile)
void push (Data, Pile)
Data top (Pile)
void pop (Pile)



File


```
struct zqueue {
    int size ;
    List in, out ; } ;
typedef struct stack * Stack ;
```

Queue create_empty_queue ()
bool is_empty_queue (Queue)
void push_back (Data, Queue)
Data top (Queue)
void pop_front (Queue)