Structures de données

- manipulations d'informations en "grand nombre"

L i c P r o S I L

Structures de données

- manipulations d'informations en "grand nombre"
- de façon raisonnée, prévisible

L i c P r o S I L

Structures de données

- de façon raisonnée, prévisible, pour

--> réutiliser des fonctions

Structures de données

- de façon raisonnée, prévisible, pour

réutiliser des fonctions

--> généricité,

--> facilité d’écriture

--> catalogues

L i c P r o S I L

Structures de données

- de façon raisonnée, prévisible, pour

réutiliser des fonctions

préparer les algorithmes

--> complexité,

--> preuve de correction

Définition algébrique

Une structure de donnée (algébrique) est définie par

- **des constructeurs**
- **des destructeurs**
- **des prédicats**
- **des axiomes**

Exemple de définition algébrique :

couples d'entiers CE

- **des constructeurs**

$$(,):\ Z\text{x}Z \longrightarrow CE$$

- **des destructeurs (observateurs)**

$$\text{gche} : CE \longrightarrow Z$$

$$\text{drte} : CE \longrightarrow Z$$

- **des prédicats (reconnaisseurs)**
- **des axiomes**

Exemple de définition algébrique :

couples d'entiers CE

- **des axiomes**

$$\forall x, y \in Z, \mathrm{gche}(x, y) = x$$

$$\forall x, y \in Z, \mathrm{drte}(x, y) = y$$

Exemple de définition algébrique :

couples d'entiers CE

- **des théorèmes**

$$\forall C \in CE, (\text{gche}(C), \text{drte}(C)) = C$$

Exemple de définition algébrique :

couples d'entiers CE

- **théorème**

$$\forall C \in CE, (\mathrm{gche}(C), \mathrm{drte}(C)) = C$$

- **preuve**

soit $C \in CE$, par définition des constructeurs,

$\exists x, y \in Z \,/\, C = (x, y)$, prenons de tels x, y

alors $(\mathrm{gche}(C), \mathrm{drte}(C)) = (\mathrm{gche}(x, y), \mathrm{drte}(x, y))$

alors $(\mathrm{gche}(C), \mathrm{drte}(C)) = (x, y)$ par les axiomes

alors $(\mathrm{gche}(C), \mathrm{drte}(C)) = C$ par def de x et y

QED

Implémentation

Une structure de donnée

est implémentée par

- **ses constructeurs**
- **ses destructeurs**
- **ses prédicats**
- **+++ la vérification des axiomes**

Exemple d'implémentation :

```
type couple=record g,d:integer end;
```

• **des constructeurs**

```
function cple(x,y:integer):couple;
var c:couple;
begin c.g:=x; c.d:=y; cple:=c end;
```

Exemple d'implémentation :

```
type couple=record g,d:integer end;
```

- **des destructeurs (observateurs)**

```
function gche(c:couple):integer;
begin gche:=c.g end;

function drte(c:couple):integer;
begin drte:=c.d end;
```

Exemple d'implémentation :

```
type couple=record g,d:integer end;
```

- **les axiomes**

$$\forall x, y \in Z, \mathrm{gche}(x, y) = x$$

$$\forall x, y \in Z, \mathrm{drte}(x, y) = y$$

deviennent

```
gche(cple(x,y))==x
drte(cple(x,y))==y
```

Exemple d'implémentation :

```
type couple=record g,d:integer end;
```

• **l'axiome**

gche(cple(x,y))==x

se prouve :

```
function cple(x,y:integer):couple;
  begin c.g:=x; c.d:=y; cple:=c end;
function gche(c:couple):integer;
  begin gche:=c.g end;
```

cple(x,y) $\longrightarrow$ c avec c.g=x

gche(cple(x,y)) $\longrightarrow$ gche(c) avec c.g=x

gche(cple(x,y)) $\longrightarrow$ c.g avec c.g=x

gche(cple(x,y)) $\longrightarrow$ x **QED**

PROCESSUS

analyse du problème à résoudre

choix d'une structure de données

utilisation d'une bibliothèque prouvée

création d'une bibliothèque

preuve de la bibliothèque

algorithme résolvant le problème

preuve de l'algo

L i c P r o S I L

Exemple de preuve :

```
type couple=record g,d:integer end;
```

- **l'algorithme spécifié par**

echg : CE $\rightarrow$ CE

et $\forall x, y \in Z, \mathrm{echg}(x, y) = (y, x)$

peut s'implémenter

```
function echg(c:couple):couple;
    begin
       echg:=cple(drte(c),gche(c))
    end;
```

Exemple de preuve :

```
type couple=record g,d:integer end;
```

- **l'algorithme spécifié par**

echg : CE → CE

et $\forall x, y \in Z, \mathrm{echg}(x,y) = (y,x)$

peut aussi s'implémenter

```
function echg(c:couple):couple;
    var res:couple;
begin
    res.g:=c.d;
    res.d:=c.g;
    echg:=res
end;
```

MAIS

pas hors de l'implémentation de couple

Catalogue

- listes typées
- listes génériques
- files
- piles
- ensembles
- arbres génériques
- arbres binaires
- arbres binaires équilibrés
- arbres rouge et noir
- tas
- ...

Listes typées (peignes) :

liste d'entiers (LE)

- **constructeurs**

  [] : $\longrightarrow$ LE

  ; : ZxLE $\longrightarrow$ LE

- **destructeurs**

  car : LE $\longrightarrow$ Z

  cdr : LE $\longrightarrow$ LE

- **exemple**

  5;6;7;8;23;[] noté [5;6;7;8;23]

Listes typées (peignes) :

liste d'entiers (LE)

- **des prédicats**

estVide : LE $\longrightarrow$ bool

- **des axiomes**

$$\forall x \in Z, \forall l \in \mathrm{LE}, \mathrm{car}(x;l) = x$$

$$\forall x \in Z, \forall l \in \mathrm{LE}, \mathrm{cdr}(x;l) = l$$

$$\mathrm{estVide}([]) = vrai$$

$$\forall x \in Z, \forall l \in \mathrm{LE}, \mathrm{estVide}(x;l) = faux$$

L i c P r o S I L

Listes typées (peignes) :

liste d'entiers (LE)

- **des axiomes**

$$\forall x \in Z, \forall l \in \mathrm{LE}, \mathrm{car}(x;l) = x$$

$$\forall x \in Z, \forall l \in \mathrm{LE}, \mathrm{cdr}(x;l) = l$$

$$\mathrm{estVide}([]) = vrai$$

$$\forall x \in Z, \forall l \in \mathrm{LE}, \mathrm{estVide}(x;l) = faux$$

- **des théorèmes**

$$\forall l \in \mathrm{LE}, \neg \mathrm{estVide}(l) \Rightarrow \mathrm{car}(l);\mathrm{cdr}(l) = l$$

L i c P r o S I L

Listes typées (peignes) :

- **spécification**

$$\text{unSur2}:\text{LE} \to \text{LE}$$

$$\text{unSur2}([]) = []$$

$$\forall x \in Z, \text{unSur2}(x;[]) = []$$

$$\forall x, y \in Z, \forall l \in \text{LE}, \text{unSur2}(x;y;l) = x;\text{unSur2}(l)$$

- **algorithme**

indépendant de l'implémentation des listes

```
function unSur2(l:LE):LE;
 begin
   if estVide(l) then unSur2:=[]
    else if estVide(cdr(l)) then unSur2:=[]
       else unSur2:=
                     cons(car(l),unSur2(cdr(cdr(l))))
 end;
```

File typées :

file d'entiers (FE)

- **constructeurs**

  [] : $\longrightarrow$ FE

  e : ZxFE $\longrightarrow$ FE

- **destructeurs**

  der : FE $\longrightarrow$ Z

  sfd : FE $\longrightarrow$ FE

- **exemple**

  e(5,e(6,e(23,[]))) noté [5;6;23]

Files typées :

file d'entiers (LE)

- **des prédicats**

estVide : FE $\longrightarrow$ bool

- **des axiomes**

$$\text{estVide}([]) = vrai$$

$$\forall x \in Z, \forall f \in \text{FE}, \text{estVide}(\text{e}(x, f)) = faux$$

$$\forall x \in Z, \text{der}(\text{e}(x, [])) = x$$

$$\forall x, y \in Z, \forall f \in \text{FE}, \text{der}(\text{e}(x, \text{e}(y, f))) = \text{der}(\text{e}(y, f))$$

Files typées :

file d'entiers (LE)

- **des axiomes**

$$\mathrm{estVide}([]) = vrai$$

$$\forall x \in \mathrm{Z}, \forall f \in \mathrm{FE}, \mathrm{estVide}(\mathrm{e}(x, f)) = faux$$

$$\forall x \in Z, \mathrm{der}(\mathrm{e}(x, [])) = x$$

$$\forall x, y \in Z, \forall f \in \mathrm{FE}, \mathrm{der}(\mathrm{e}(x, \mathrm{e}(y, f))$$

$$\forall x \in Z, \mathrm{sfd}(\mathrm{e}(x, [])) = []$$

$$\forall x, y \in Z, \forall f \in \mathrm{FE}, \mathrm{sfd}(\mathrm{e}(x, \mathrm{e}(y, f)) = \mathrm{e}(x, \mathrm{sfd}(\mathrm{e}(y, f)))$$

Piles typées :

piles d'entiers (PE)

- **constructeurs**

  [] : $\longrightarrow$ PE

  psh : ZxPE $\longrightarrow$ PE

- **destructeurs**

  top : PE $\longrightarrow$ Z

  pop : PE $\longrightarrow$ PE

- **exemple**

  psh(5,psh(6,psh(23,[])))

Piles typées :

piles d'entiers (LE)

- **des prédicats**

estVide : PE $\longrightarrow$ bool

- **des axiomes**

$$\mathrm{estVide}([]) = vrai$$

$$\forall x \in Z, \forall p \in \mathrm{PE}, \mathrm{estVide}(\mathrm{psh}(x,p)) = faux$$

$$\forall x \in Z, \forall p \in \mathrm{PE}, \mathrm{top}(\mathrm{psh}(x,p)) = x$$

$$\forall x \in Z, \forall p \in \mathrm{PE}, \mathrm{pop}(\mathrm{psh}(x,p)) = p$$

Piles typées :

piles d'entiers (PE)

- **remarques**

-- ressemblance avec les listes
-- implémentation par procédures

- **constructeur/destructeurs**

faitPileVide(p) *{p=[ ]}*
push(x,p) *{p=push(x,p)}*
pop(x,p) *{x=top(p);p=pop(p)}*
estPileVide(p)

L i c P r o S I L

Arbres binaires d'entiers : ABE

- **constructeurs**

  • : → ABE

  (/ \ ) : ABExZxABE → ABE

- **destructeurs**

  etq : ABE → Z

  gche : ABE → ABE

  drte : ABE → ABE

- **exemples**

  (•/5\•) ((•/12\(•/3\•))/21\(•/7\•))

Arbres binaires d'entiers : ABE

- **prédicats**

estVide : ABE $\longrightarrow$ bool

- **axiomes**

$$\forall n \in Z, \forall g, d \in \text{ABE}, \text{etq}(g / n \backslash d) = n$$

$$\forall n \in Z, \forall g, d \in \text{ABE}, \text{gche}(g / n \backslash d) = g$$

$$\forall n \in Z, \forall g, d \in \text{ABE}, \text{drte}(g / n \backslash d) = d$$

$$\text{estVide}(\bullet) = vrai$$

$$\forall n \in Z, \forall g, d \in \text{ABE}, \text{estVide}(g / n \backslash d) = faux$$

L i c P r o S I L

Tas d'entiers :

tas

- **constructeurs**

tasVide : $\rightarrow$ tas

entasser : Zxtas $\rightarrow$ tas

- **destructeurs**

racine : tas $\rightarrow$ Z

sansRacine : tas $\rightarrow$ tas

- **axiomes**

un tas est un arbre binaire étiqueté vérifiant qu'un père est toujours plus grand que ses fils