Programmes LOOP : variables $x, y, \ldots$

(res $\hat{=}$ variable "résultat")

Instructions :

$x := c$ $\quad c \in \mathbb{N}$ $\quad$ SKIP

$y := x + c$, $y := x - c$

LOOP(x) DO P OD

$add(x,y) = x + y$

$$sgn(x) = \begin{cases} 1 & \text{si } x > 0 \\ 0 & \text{si } x = 0 \end{cases}$$

$$x \dot{-} y = \begin{cases} x - y & \text{si } x \geq y \\ 0 & \text{sinon} \end{cases}$$

(différence "tronquée", à ne pas confondre avec $x - y$ dont le résultat peut être négatif)

Quelques programmes LOOP :

① $x \dot{-} y$

```
LOOP(y) DO x := x-1 OD
res := x
```

② $sgn(x) = \begin{cases} 1 & \text{si } x > 0 \\ 0 & \text{si } x = 0 \end{cases}$

```
y := 1 ;
LOOP(x) DO y := y-1 OD ;
// y = { 1 si x = 0
//     { 0 sinon
res := 1 ∸ y
```

③ Que calcule le programme suivant ?

```
LOOP(y) DO x := x-1 OD        x ∸ y
z := y
LOOP(x) DO z := z+1 OD        y + (x ∸ y)
res := z
```

$x \geq y$: x ; $<$: y

$\max(x, y)$

(25)

```
y := 0;
LOOP (x) DO y := 1 OD
res := y
```

Si : $x = 0$: y reste 0

$x > 0$: y devient 1

(4) $test(x,y) = x * (1 \dot{-} y)$

$$1 \dot{-} y = \begin{cases} 1 & \text{si } y \geq 0 \\ 0 & \text{si } y > 0 \end{cases}$$

$$x * (1 \dot{-} y) = \begin{cases} x & \text{si } y = 0 \\ 0 & \text{si } y > 0 \end{cases}$$

(5) $c > 0$ Constante

$x_0 := 0$;
$x_1 := 1$;
⋮
$x_{c-1} := c-1$;

ex. $c = 3$

$x \in x_0$

$x_2 \longrightarrow x_1$

Loop(x) DO
  $x := x_0$;
  $x_0 := x_1$;
  $x_1 := x_2$;
  ⋮
  $x_{c-2} := x_{c-1}$;
  $x_{c-1} := x$;
OD
res := x_0

x mod 3

⑥

$x_0 := 0$;

$x_1 := 1$;

⋮

$x_{c-1} := c-1$;

loop(x) DO

$x := x_c$;

$x_c := x_{c-1}$;

⋮

$x_1 := x_0 + 1$;

$x_0 := x$

OD

res := x_0

x div 3

0 div 3 = 0

1

2

3 div 3 = 1

⋮

⑦ $\lceil \sqrt{n} \rceil$

$m = \lceil \sqrt{n} \rceil \iff$

m = le plus petit entier tq. $n \leq m^2$

$(m-1)^2 < n \leq m^2$

Calculer $0^2, 1^2, 2^2, 3^2 \ldots, n^2$

et comparer avec n

Bien-sûr on peut utiliser une boucle while pour trouver le + petit m tq. $n \leq m^2$, mais on peut aussi utiliser une boucle LOOP pour faire ça, car on sait que ce m ne peut pas dépasser n.

m := 0

LOOP (n) DO

- comparer m^2 et n
- si $m^2 < n$: incrémenter m

sinon : garder ce résultat m, sans continuer à incrémenter m

OD

On peut utiliser une variable de plus, disons B (break), au début 0, qui passe à 1 quand $n \leq m^2$

B = 0 → incrémenter m

B = 1 → m ne change pas

- Programmes WHILE :

  on rajoute la boucle WHILE :

  WHILE $(x \neq 0)$ DO P OD

- Programmes GOTO :

  on remplace les boucles LOOP, WHILE par des sauts :

  IF $(x = 0)$ THEN GOTO ℓ FI

  GOTO ℓ

Les programmes WHILE et GOTO calculent les mêmes fonctions.

Il y a des fonctions qui sont WHILE-calculables, mais pas LOOP-calculables

De WHILE à GOTO

On peut exprimer une boucle WHILE par programme GOTO :

WHILE (x≠0) DO P OD équivaut à

1: IF (x=0) THEN GOTO 4 FI

2: P;

3: GOTO 1

4:

Rq. On peut traduire les boucles LOOP de la même façon.

Rq. Comme on a déjà enrichi les programmes LOOP avec IF-THEN-ELSE on peut enrichir les programmes WHILE avec des conditions utilisant les

opérateurs booléens $\wedge, \vee, \neg$ dans les conditions de boucles (ou dans les conditions de LOOP, IF ...)

$f, g : \{0,1\}^k \to \{0,1\}$

$0 \hat{=}$ Faux, $1 \hat{=}$ Vrai

$f \wedge g$, $f \vee g$, NOT f seront aussi calculables, dans le même modèle que f, g

$f(x_1, \ldots, x_k)$, $g(x_1, \ldots, x_k)$

$f \wedge g = f * g$

$0 * 0 = 0 = 0 * 1 = 1 * 0$

$1 * 1 = 1$

$f \vee g = \operatorname{sgn}(f + g)$

$\text{not } f = 1 \dot{-} f$

WHILE $(x \neq 0 \wedge y > 0)$ DO P OD

$\Downarrow$

$z \leftarrow (x = 0) \wedge (y > 0)$ ⊛

$\in \{0,1\}$

WHILE $(z \neq 0)$ DO P ; ⊛ OD

De GOTO à WHILE

On montre comment simuler un programme du type

$P = I_1 ; I_2 ; \ldots I_m$

$\|$

HALT

On utilise une nouvelle variable PC (program counter) qui mémorise le numéro de l'instruction actuelle.

On suppose que chaque instruction

$I_j \longrightarrow$ programme while P_j qui la simule

$PC := 1$

WHILE $(PC \neq m)$ DO

IF $(PC = 1)$ THEN P_1 FI ;

IF $(PC = 2)$ THEN P_2 FI ;

⋮

IF $(PC = m-1)$ THEN P_{m-1} FI

OD

Comment on déf $P_1, \ldots, P_{m-1}$?

- Si $I_j = (y := x \pm c$, ou $:= c)$

alors $P_j = [I_j ;\ PC := PC + 1]$

- Si I_j = GOTO ℓ , alors

$$P_j = [\ PC := \ell\ ]$$

- Si I_j = IF $(x \neq 0)$ THEN GOTO ℓ FI

$$P_j = [\ PC := PC + 1\ ;$$
$$\text{LOOP}(x)\ \text{DO}\ \ PC := \ell\ \text{FI}\ ]$$

Si I_j = IF $(x = 0)$ THEN GOTO ℓ FI

$$P_j = [\ PC' := PC + 1\ ;$$
$$PC := \ell\ ;$$
$$\text{LOOP}(x)\ \text{DO}\ \ PC := PC'\ \text{OD}\ ]$$