Maîtrise d'informatique
Module de Projet 1

Examen – Compilateur Pascal

Philippe MARQUET

Janvier 2000

Documents autorisés — Durée : 2 heures

Soit PPx une extension de PP2 (chapitre 6, Variables de type tableau) introduisant une nouvelle entité : les *indicateurs de situation* [Zahn, 1974], illustrés par l'exemple de la figure 1.

PPx comporte une structure de contrôle supplémentaire : la construction **until**. Elle se dérive sous deux formes : la boucle **loop until** et la construction **until** simple dont les formes générales sont :

loop until $\langle \textit{situation} \rangle_1$ **or** ... **or** $\langle \textit{situation} \rangle_n$:
    $\langle \textit{statement list} \rangle_0$;
repeat ;
with $\langle \textit{situation} \rangle_1 \longrightarrow \langle \textit{statement list} \rangle_1$;
    ⋮
    $\langle \textit{situation} \rangle_n \longrightarrow \langle \textit{statement list} \rangle_n$;
end ;

until $\langle \textit{situation} \rangle_1$ **or** ... **or** $\langle \textit{situation} \rangle_n$:
    $\langle \textit{statement list} \rangle_0$;
with $\langle \textit{situation} \rangle_1 \longrightarrow \langle \textit{statement list} \rangle_1$;
    ⋮
    $\langle \textit{situation} \rangle_n \longrightarrow \langle \textit{statement list} \rangle_n$;
end ;

PPx comporte aussi une instruction $\langle \textit{situation} \rangle$ qui signifie que la situation donnée est apparue. Une telle instruction n'est autorisée que dans une construction **until** qui a déclarée cette situation.

La sémantique de la construction **until** est d'exécuter les instructions de $\langle \textit{statement list} \rangle_0$ jusqu'à ce qu'apparaisse une des situations. Les instructions de la $\langle \textit{statement list} \rangle_i$ correspondante sont alors exécutées.

Exercice 1 (Grammaire)
Donner la grammaire de PPx extension de celle de PP2.

Exercice 2 (Sémantique)
Préciser la sémantique de la construction **until** simple lorsque l'on atteint la fin de $\langle \textit{statement list} \rangle_0$ sans qu'aucune des situations n'ait apparue.

Exercice 3 (Analyse syntaxique)
Donner une procédure d'analyse syntaxique pour la construction **until** générale.

Exercice 4 (Alternatives)

– Discuter (brièvement) de l'alternative suivante au deux constructions **until** : introduction de la seule construction **until** simple et introduction d'une boucle sans fin **loop**...**repeat**.
– Discuter (brièvement) de l'utilité du mot clé **repeat** dans la boucle **loop until**.

Exercice 5 (Schéma de génération de P-Code)
Donner le schéma de P-Code à générer pour une construction **until** et pour une instruction $\langle \textit{situation} \rangle$.

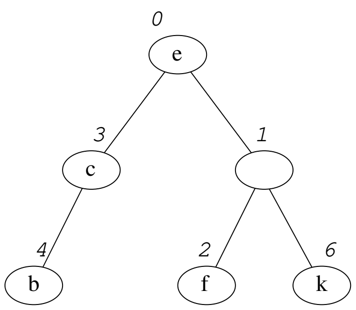
Exercice 6 (Table des symboles)
Donner les informations devant être associées à un indicateur de situation dans la table des symboles.

Id : xam00.tex,v 1.4 2001/11/02 13 :14 :14 marquet Exp

```
const SIZE = 127,
      BOTTOM = -1 ;
var x, i,
    root, next,
    A [SIZE], L [SIZE], R [SIZE] ;

begin
  loop until left_leaf or right_leaf :
    if x < A[i] then
      if L[i] <> BOTTOM then i := L[i] else left_leaf ;
    else
      if R[i] <> BOTTOM then i := R[i] else right_leaf ;
  repeat
  with
    left_leaf  -> L[i] := next ;
    right_leaf -> R[i] := next ;
  end ;
  A[next] := x ; L[next] := BOTTOM ; R[next] := BOTTOM ;
  next := next + 1 ;
end ;
```

FIG. 1 – Exemple d'utilisation des indicateurs de situation.
Un arbre binaire de recherche est codé dans les tableaux `A` (qui détient les valeurs), `L` (qui contient l'indice dans les tableaux du fils gauche) et `R` (fils droit). La variable `next` est le prochain emplacement libre dans les tableaux. La variable `root` est l'indice de la racine de l'arbre. La valeur `BOTTOM` pour `L` ou `R` dénote un fils non existant. (Voir figure 2.)
`left_leaf` et `right_leaf` sont deux indicateurs de situation.
Ce code insère la valeur `x` dans l'arbre de racine `i` (on suppose `x` non présent dans l'arbre).



	0	1	2	3	4	5	6
A	e	g	f	c	b	k	–
L	**3**	**2**	**–1**	**4**	**–1**	**–1**	**–1**
R	**1**	**5**	**–1**	**–1**	**–1**	**–1**	**–1**

FIG. 2 – Représentation d'un arbre binaire de recherche par des tableaux.

Exercice 7 (Erreurs)
Donner une liste des erreurs pouvant survenir dans une construction **until**. Expliquer comment le compilateur peut détecter ces erreurs.

Exercice 8 (Génération de code)
Étendre la procédure d'analyse syntaxique de la question 3 pour y inclure les instructions de génération de P-Code.