Maîtrise d'informatique
Module de Projet 1

Examen – Compilateur Pascal

Philippe MARQUET

Janvier 2003

Documents autorisés — Durée : 2 heures

Soit PPx une extension de PP2 (chapitre 6, Variables de type tableau) introduisant la nouvelle structure de contrôle suivante :

```
INST      ::=  INSTS | AFFEC | SI | TANTQUE | ECRIRE | LIRE | TANTQUUN
TANTQUUN  ::=  whileone COND do INST { ; COND do INST } end
```

L'instruction `whileone` est une boucle formée d'une suite d'instructions gardées ; chaque instruction gardée est constituée d'une instruction précédée d'une condition, la garde.

L'exécution d'une itération de la boucle est réalisée ainsi : si une au moins une des gardes est évaluée à vraie, une garde est alors choisie parmi celle évaluées à vraie et l'instruction correspondante est exécutée tant que sa garde est vraie. Quand cette garde passe à faux, l'itération courante de la boucle `whileone` se termine. L'exécution de la boucle `whileone` se poursuit jusqu'à ce que toutes les gardes soient fausses.

Cette instruction est illustrée par l'exemple de la figure 1.

```
const SIZE = 1023 ;
var i, j,
    n, m,
    v,
    tmp,
    A [SIZE] ;

v := A[m] ;
i := n ; j := m ;

whileone
  A[i] < v  do i := i+1 ;
  v <= A[j] do j := j-1 ;
  A[j] < A[i] do begin
    tmp := A[i] ; A[i] := A[j] ; A[j] := tmp
  end
end ;
```

FIG. 1 – Exemple d'utilisation de la structure `whileone`
Les éléments du tableau `A` de `A[n]` à `A[m]` sont réarrangés pour être partitionnés en deux parties : la partie de gauche `A[n]` à `A[j-1]`, pour un certain `j`, qui contient tous les éléments inférieurs à un `v` donné ; la partie droite `A[j]` à `A[m]` qui contient tous les éléments plus grands que ou égaux à `v`.

Id: xam03.tex,v 1.5 2003/01/19 09:00:27 marquet Exp

Exercice 1 (Analyse syntaxique)
Donner une procédure d'analyse syntaxique pour l'instruction `whileone`. □

Exercice 2 (Alternative syntaxique)
Discuter brièvement de l'alternative syntaxique suivante à l'instruction `whileone` présentée :

```
TANTQUUN  ::=  whileone while COND do INST { orwhile COND do INST }
```

□

Exercice 3 (Schéma de génération de P-Code)
Donner le schéma de P-Code à générer pour une instruction `whileone`.

On pourra utiliser une variable gérée par le compilateur pour marquer le fait d'être entré ou non dans une des boucles du `whileone`. On s'assurera que cette variable ne soit pas partagée entre différentes instances imbriquées de `whileone`. On pourra étendre le P-Code pour faciliter la gestion de cette variable. □

Exercice 4 (Traduction en P-Code)
Traduire en P-Code le programme de la figure 1. □

Exercice 5 (Génération de code)
Étendre la procédure d'analyse syntaxique de l'exercice 1 pour y inclure les instructions de génération de P-Code. □

Fin du sujet