Maîtrise d'informatique
Module de Projet 1

Examen – Compilateur Pascal

Philippe Marquet

Janvier 1995

Documents autorisés — Durée : 2 heures

Ce contrôle est divisé en deux problèmes indépendants.

1 Échappement dans une boucle `while`

On introduit deux nouvelles instructions permettant des échappements dans les boucles `while` : le `break` et le `continue`.

La grammaire de PP1 (page 26 du poly) est donc modifiée ainsi :

```
TANTQUE   : :=  while COND do INST
INST      : :=  ... | break | continue
```

La sémantique de ces instructions est la suivante :
- une instruction `continue` termine l'itération courante de la boucle `while` ;
- une instruction `break` termine la boucle `while`.

Dans le cas de boucles `while` imbriquées, la boucle la plus imbriquées est considérée.

Exercice 1
Donner le schéma de P-Code à générer pour la boucle suivante :

```
while <cond> do begin
  <inst1>
  ... continue
  <inst2>
  ... break
  <inst3>
end
```

Exercice 2
Suivant la construction du compilateur définie dans le poly, quelles informations doivent être partagées entre la procédure `TANTQUE` (poly page 49) et les procédures `CONTINUE` et `BREAK` d'analyse des instructions `continue` et `break` ? Comment proposez vous de conserver ces informations. Assurez vous de considérer les boucles `while` imbriquées, en particulier sur des codes sources tels celui-ci :

```
while <cond1> do begin
  while <cond2> do begin
    ... break
    ... continue
  end
  ... break
  ... continue
```

```
    while <cond3> do begin
      ... break
      ... continue
    end
  end
```

Exercice 3
Expliciter les modifications nécessaires à la procédure `TANTQUE` (poly page 49) pour prendre en compte ces nouvelles instructions d'échappement.

Exercice 4
Donner le code des procédures `CONTINUE` et `BREAK`.

Exercice 5 (Bonus/Question facultative)
Proposer une syntaxe et une sémantique permettant de faire des échappements à tout niveau dans les boucles `while` imbriquées.

2 Incrémentation

On se propose d'ajouter deux opérateurs d'incrémentation et de décrémentation au langage PP2 (c'est-à-dire l'extension de PP1 autorisant la définition de tableaux à une dimension ; chapitre 7, page 61). On modifie donc la grammaire :

```
INST   : :=  ... | INCR | DECR
INCR   : :=  ID ++
DECR   : :=  ID --
```

On peut donc écrire

```
I++
T[X]--
```

à la place de

```
I := I + 1
T[X] := T[X] - 1
```

Exercice 6
Donner le schéma de P-Code à générer pour une de ces deux instructions. Écrire l'une des deux procédures `INCR` et `DECR` d'analyse et de génération de code de ces nouvelles instructions **dans le langage intermédiaire du poly** (page 19).

Exercice 7
On étend le langage intermédiaire du poly par une instruction `DUP` qui duplique le sommet de pile. Donner le nouveau schéma de P-Code à générer pour les instructions `INCR` ou `DECR`.

Exercice 8
Pourquoi est-ce intéressant d'étendre le langage intermédiaire de cette façon ?