USTHB
Faculté d'Electronique et d'Informatique
Département d'Informatique
Année Universitaire 2008/2009
1 ère année Master Ingénierie des Logiciels

Durée :1H 30
Mai 2009

Examen Final
Compilation 2

Exercice 2 : (8 pts)
Considérez le programme en C suivant :

```
Main()
{
    short input=1 ;    short b=1;    short a,c,d;    int
mult,div,result ;
  while(1)
 { b=b*b;
    switch (input) {
      case 1:
            { mult=a*b;
              div=c/d;
              a=c/d+a*b;
              input++;}
      case 2:
            { mult=a*b;
              div=c/d;
              a=c/d+a*b;
              input++;
              break; }
      case 3:
            { div=c/d;
              input++;
              exit(0);}
      default: {
              a=c/d+a*b;
              input++; }
    } }
 return(0);
}
```

1- Produisez le code intermédiaire correspondant sous forme de quadruplés.
2- Optimisez le code obtenu en (1).
3- Générez le code objet à partir du code obtenu en (2), en vous basant sur les instructions Assembleur utilisées dans le TP .

Exercice 2 :(12 pts)

Soit le prototype de programme suivant : dynamique.

```
PROGRAM nomprog;
      /*déclarations*/
      ..............
 Procedure nomproc(paramètres formels)
  BEGIN
      ..................
  END;
      /*programme principal*/
BEGIN
      ..................
     CALL nomproc(paramètres effectifs);
      ..................
END.
```

- Dans le cas d'une analyse ascendante, écrivez les routines sémantiques relatives à :
  - a- La déclaration de procédure,
  - b- L'appel de procédure,
  - c- Le retour au programme appelant.

'co2: intermediate ophmise' (3pts)

1 - (:=, 1, , input)

2 - (:=1, , b)

3 - (/, c, d, T1)

4 - (:=, T1, , div)

5 - (BNE, (12), input, 1)

6 - (+, a, b, , T2)

7 - (:=, T2, , nuLT)

8 - (+, div, nuLT, T3)

9 - (:=, T1, , a)

10 (+, input, 1, T4)

11 (:=, T4, , input)

12 (BNE, (20), input, 2)

13 (+, a, b, T5)

14 - (:=, T5, , nuLT).

15 - (+, div, nuLT, T6)

16 (:= T6, , a)

17 - (+, input, 1, T7)

18 - (:=, T7, , input)

19 - (BR, (29), ,)

20 (BNE, (24), input, 3)

21 (+, input, 1, T8)

22 - (:=, T8, , input)

23 - (BR, (30), ,)

24 - (+, a, b, T9)

25 - (+, DIV, T9, T10)

26 - (:=, T10, , a)

27 - (+, input, 1, T11)

28 - (:=, T11, , input)

29 - (BR, (5), ,)

30 - END

) Mov Input, 1 (j'ai accepté Mov AX, 1
) Mov b, 1 Mov Input, AX)
) Mov AX, c
) Div AX, d
) Mov Div, AX
) etiq1: Comp INput, 1
) JNZ etiq1. (autre JNE)
) Mov AX, a
) Mul AX, b
) Mov Mult, AX
) ADD AX, Div
) Mov a, AX
) Mov INPut, Ax
) ADD AX, 1
) Mov input, AX
) etiq1: comp input, 2
) JNZ etiq2
) Mov AX, a
) Mul AX, b
) Mov Mult, AX
) ADD Div, AX
) Mov a, AX
) Mov AX, input
4) ADD AX, 1
5) Mov input, AX

6) JMP etiq3

7) etiq2: Comp input, 3

8) JNZ etiq4

) Mov AX, input

) ADD AX, 1

) Mov input, AX

) JMP etiq5

)etiq4 Mov AX, e

) ADD AX, b

~~Mov~~ /a

) ADD AX, div

) Mov ~~AX~~ a, AX

) Mov AX, input

) ADD AX, 1

) Mov ~~AX~~ input, AX

) etiq3 : JMP etiq6

) etiq5: (fi)

grammaire syntaxique correspondant à la déclaration :

- < Decl-proc > → Procedure < nom-proc > (< liste-param formel >)
- < liste-param-formels > → param 1, < liste-param-formels >
- < liste-param-formels > → < liste-param-formels >, param 1

Cas ascendant

- grammaire syntaxique correspondant à l'appel :

- < appel-Proc > → call < nom-proc > (< liste-param effectifs >) ;
- < liste-param-effectifs > → param, < liste-param-effectifs > 0,5

- retour

0,25 < retour-proc > → return ;

Cas ascendant, la récursivité doit être à gauche.

Transformation de la grammaire, afin d'inclure les routines sémantiques :

0,25

- < Decl-proc > → A < liste-param-formels > ;
- < A > → Procédure < nom-proc > ⑤ ↑①
- < liste-param-formels > → < B >, param 1 ↑③
- < B > → < liste-param-formels > ↑②
- < Appel-Proc > → < C > (< liste-param-effectifs >)
- < C > → call < nom-proc > ↑③
- < liste-param-effectifs > → < liste-param-effectifs > ④, < param > ↑④
- < retour-proc > → return ; ⑤ [A la fin de la]

/* ... pendant l'exécution */

lookup (id.nom, p)

Si p=0 alors ('Erreur : id n'existe pas')

Sinon - Sauvegarder le nom → Sauv.nom

nb param := 0;

2 { - empiler (pile ds données, display de la procédure)

- " (pile ds donné, tête pile procédure)

- ~~(pile ds données, param implicite)~~

- achtarea ← @ 2D de la procédure

fsi

Routine ② /* Cette routine est appelée autant de fois qu'il y a de paramètres formels */

lookup (id.nom, p)

Si p=0 alors 'Erreur : id n'existe pas'

Sinon Compter le nombre de paramètres

nb param := nb param + 1

Sauv-type := p-type

1 /* Sauvegarde du type du paramètre formel pour comparer avec paramètre effectif */

empiler (pile ds données, param-formel)

fsi

Routine (3) /* l'appel à la procédure, vérifier si c'est le même nom */

```
lookup (id.nom, p)
nb param 1 := 0
Si p = 0 alors ('Erreur id n'existant')
        Sinon  Si  p.nom ≠ Sauv.nom
               alors ('erreur : ce n'est pas le même
                      nom de la procédure')
               Sinon  /* Commencer à comparer
                         les paramètres et associer
                         param effectif à paramètre
                         formel */
               fsi
fsi
```

Routine (4)

```
lookup (typ.id.nom, p)
Si p = 0 alors 'Erreur : id n'existant')
        Sinon  Si p.type ≠ Sauv.type
               alors ('erreur : les types des 2
                      paramètres sont pas compatibles')
               Sinon nb.param 1 := nbparam + 1
               fsi  générer association paramètre
                    formel, paramètre effectif
                    remplacer par les valeurs */
fsi
```

Routine (4')

```
/* nous sommes au dernier paramètre
   effectif */
Si nb param ≠ nb param 1
```

formels ")

- Sauvegarder l'adresse de retour, parmi les paramètres
- Générer un branchement vers le début de la procédure.

Routine (5) /* Cette routine est appelée juste après le ; qui suit la liste des paramètres effectifs */

(générer) - réservation du contexte, en particulier l'adresse de retour de la procédure appelante.

(générer) - libération de la ZD de la procédure

2- activation ← de ZD de la procédure appelante

- générer un branchement vers l'adresse de retour.