Universidade Federal de Alfenas - UNIFAL-MG
Bacharelado em Ciência da Computação
Prof. Luiz Eduardo da Silva

Trabalho de Compiladores
Função com retorno

Objetivo

O objetivo desse trabalho é modificar o projeto do compilador para linguagem simples a fim de permitir a compilação de funções no estilo da linguagem C, com múltiplos pontos de saída da função, através do comando *retorne*.

Problema

A função, nas linguagens de programação, é um pedaço de código que pode ser invocado em qualquer lugar do programa com o uso do nome da função. Valores podem ser passados através dos parâmetros e o resultado da função é devolvido na expressão em que a função foi chamada. Linguagens de programação no estilo de Pascal (Pascal-like) usam um mecanismo de retorno do resultado através da variável-nome-da-função, conforme foi apresentado nos exemplos da linguagem Simples. A atribuição, dentro da função, para a variável-nome-da-função define o valor que será retornado da chamada da função, quando a função é finalizada.

As linguagens C-like implementam um mecanismo diferente. Existe o comando *return expressao* que pode ser colocado em qualquer ponto da função e determina sua saída com o valor da expressão. Diferente do mecanismo Pascal-like, nas linguagens parecidas com C é possível múltiplos pontos de saída da função.

```
1 int f(int a, int b) {
2   if (a > b)
3     return a;
4   return b;
5 }
```

Roteiro

1. Defina regras para permitir a declaração de funções, conforme o seguinte exemplo:

```
1 func inteiro MAIOR (inteiro A inteiro B)
2 inicio
3     se A > B
4         entao retorne A
5         senao retorne B
6   fimse
7 fimfunc
```

2. Defina a palavra-reservada *retorne*, no léxico, e um comando *retorne expressao*, no sintático, que só pode ser usado no contexto da função (em C isso não é necessário, pois o corpo principal do código está numa função denominada *main*).

3. Considere que a função só admite parâmetros passados por valor.

4. O compilador deve verificar compatibilidade no tipo da função, no número e tipo dos parâmetros na chamada da função.

5. O compilador deve ser capaz de produzir as traduções conforme os seguintes exemplos:

Exemplo 1:

```
1          programa teste1
2          func inteiro maior (inteiro a inteiro b)
3          inicio
4            se a > b
5               entao retorne a
6               senao retorne b
7            fimse
8          fimfunc
9          inicio
10            escreva maior (10 20)
11         fimprograma
```

```
1          INPP
2          DSVS    L0
3  L1      ENSP
4          CRVL    -4
5          CRVL    -3
6          CMMA
7          DSVF    L2
8          CRVL    -4
9          ARZL    -5
10         RTSP    2
11         DSVS    L3
12 L2      NADA
13         CRVL    -3
14         ARZL    -5
15         RTSP    2
16 L3      NADA
17 L0      NADA
18         AMEM    1
19         CRCT    10
20         CRCT    20
21         SVCP
22         DSVS    L1
23         ESCR
24         FIMP
```

Exemplo 2:

```
1          programa teste2
2          inteiro x y
3          func inteiro maior (inteiro a inteiro b)
4          logico c
5          inicio
6            c <- a > b
7            se c
8              entao retorne a
9              senao retorne b
10           fimse
11         fimfunc
12         inicio
13           leia x
```

```
14            leia y
15            escreva maior (x y)
16          fimprograma
```

```
1           INPP
2           AMEM    2
3           DSVS    L0
4   L1      ENSP
5           AMEM    1
6           CRVL    -4
7           CRVL    -3
8           CMMA
9           ARZL    0
10          CRVL    0
11          DSVF    L2
12          CRVL    -4
13          ARZL    -5
14          DMEM    1
15          RTSP    2
16          DSVS    L3
17  L2      NADA
18          CRVL    -3
19          ARZL    -5
20          DMEM    1
21          RTSP    2
22  L3      NADA
23  L0      NADA
24          LEIA
25          ARZG    0
26          LEIA
27          ARZG    1
28          AMEM    1
29          CRVG    0
30          CRVG    1
31          SVCP
32          DSVS    L1
33          ESCR
34          DMEM    2
35          FIMP
```

Exemplo 3:

```
1           programa teste3
2           inteiro x
3           func inteiro quadrado (inteiro a)
4           inicio
5             retorne a * a
6           fimfunc
7           inicio
8             leia x
9             escreva quadrado (quadrado (x))
10          fimprograma
```

```
1           INPP
2           AMEM    1
3           DSVS    L0
4   L1      ENSP
5           CRVL    -3
6           CRVL    -3
7           MULT
```

```
8           ARZL    -4
9           RTSP    1
10  L0      NADA
11          LEIA
12          ARZG    0
13          AMEM    1
14          AMEM    1
15          CRVG    0
16          SVCP
17          DSVS    L1
18          SVCP
19          DSVS    L1
20          ESCR
21          DMEM    1
22          FIMP
```

Exemplo 4:

```
1           programa teste4
2           inteiro x
3           func inteiro fatorial (inteiro a)
4           inicio
5             se a > 0
6                entao retorne a * fatorial (a - 1)
7                senao retorne 1
8             fimse
9           fimfunc
10          inicio
11            leia x
12            escreva fatorial (x)
13          fimprograma
```

```
1           INPP
2           AMEM    1
3           DSVS    L0
4   L1      ENSP
5           CRVL    -3
6           CRCT    0
7           CMMA
8           DSVF    L2
9           CRVL    -3
10          AMEM    1
11          CRVL    -3
12          CRCT    1
13          SUBT
14          SVCP
15          DSVS    L1
16          MULT
17          ARZL    -4
18          RTSP    1
19          DSVS    L3
20  L2      NADA
21          CRCT    1
22          ARZL    -4
23          RTSP    1
24  L3      NADA
25  L0      NADA
26          LEIA
27          ARZG    0
28          AMEM    1
29          CRVG    0
```

```
30          SVCP
31          DSVS    L1
32          ESCR
33          DMEM    1
34          FIMP
```

Entrega

1. Incluir um comentário no cabeçalho de cada programa fonte com o seguinte formato:

```
/*+-------------------------------------------------------------------
  |            UNIFAL - Universidade Federal de Alfenas.
  |              BACHARELADO EM CIENCIA DA COMPUTACAO.
  | Trabalho..: Funcao com retorno
  | Disciplina: Teoria de Linguagens e Compiladores
  | Professor.: Luiz Eduardo da Silva
  | Aluno.....: Fulano da Silva
  | Data......: 99/99/9999
  +-------------------------------------------------------------------*/
```

2. A pasta com o projeto deverá incluir o seguinte arquivo *makefile*:

```
simples : utils.c lexico.l sintatico.y;\
        flex -o lexico.c lexico.l;\
        bison -o sintatico.c sintatico.y -v -d;\
        gcc sintatico.c -o simples

limpa   : ;\
        rm -f lexico.c sintatico.c sintatico.output *~ sintatico.h simples\
```

3. O compilador deverá ter o nome "simples"e executado através da seguinte chamada

```
./simples nomeprograma[.simples]
```

4. Enviar num arquivo único (.ZIP), a pasta do projeto com somente os arquivos fontes (lexico.l, sintatico.y, utils.c e makefile), através do Envio de Arquivo do MOODLE.