Avaliação 2 - Compiladores

Professor: Lucas Pupulin Nanni

- William D Costa - RA 89239
- Mateus Soares - RA

Todos os programas foram traduzidos para linguagem da MEPA pelo compilador desenvolvido como trabalho da disciplina

1 - Programa contendo laço de repetição e estrutura condicional aninhados:

O Objetivo deste programa, é somar os números naturais menores que *n*, onde *n < 100*. Ex:

```
f(n) = 0 + 1 + 2 + ... + n
onde n < 100

f(3) = 0 + 1 + 2 + 3
f(3) = 6
```

Código Rascal:

```
program prova1;

var x, i, result: integer;

begin
    read(x);
    i := 1;
    if(x <= 100) then
    begin
        while(i <= x) do
          begin
              result := result + i;
              i := i + 1;
          end
    end
    write(result);
end.
```

Tabela de Simbolos

Variável	Endereço
x	0, 0

Variável	Endereço
i	0, 1
result	0, 2

Compilação para código MEPA:

```
        INPP
        AMEM    3           var x, i, result: integer;
        DSVS    R0
R0:     NADA
        LEIT
        ARMZ    0, 0        read(x);
        CRCT    1
        ARMZ    0, 1        i := 1;
        CRVL    0, 0
        CRCT    100
        CMEG                x <= 100 (Condição 'if')
        DSVF    L0          'if' (desvia se 'if' = falso)
L1:     NADA                rótulo inicial 'while'
        CRVL    0, 1
        CRVL    0, 0
        CMEG                i <= x (Condição 'while')
        DSVF    L2          Desvia caso condição 'while' = falso
        CRVL    0, 2
        CRVL    0, 1
        SOMA                result + i
        ARMZ    0, 2        result := result + i;
        CRVL    0, 1
        CRCT    1
        SOMA                i + 1;
        ARMZ    0, 1        i := i + 1;
        DSVS    L1          Desvia para o início do 'while'
L2:     NADA                Fim 'while'
L0:     NADA                Fim 'if'
        CRVL    0, 2
        IMPR                write(result);
        DMEM    3
        PARA
```

Execução

```
(base) [mtuser@cp5858 mepa](main)$ ./mepa -r -i ../prova1.mepa
----------------------------------------
 Arquivo de entrada: ../prova1.mepa
 Modo Interativo:    nao
 Iniciando Execução
----------------------------------------
(entrada):50
```

```
saida: 1275
execução finalizada
```

2 - Programa contendo a definição e chamada de um procedimento com passagem de ao menos um parâmetro:

Procedimento para cálculo Fatorial.

Código Rascal:

```
program prova2;

var x: integer;

procedure fat(a: integer);
  var result:integer;
  begin
    result := 1;
    while(a > 1) do
      begin
          result := result * a;
          a := a - 1;
      end
      write(result);
  end

begin
  read(x);
  fat(x);
end.
```

Tabela de Simbolos

Variável	Endereço
x	0, 0
fat(...)	R1
result	1, 0
a	1, -4

Compilação para código MEPA:

```
        INPP
        AMEM    1           var result:integer;
        DSVS    R0
```

```
R1:     ENPR    1           procedure fat(a: integer);
        AMEM    1           var result:integer;
        CRCT    1
        ARMZ    1, 0        result := 1;
L0:     NADA                rótulo inicial 'while'
        CRVL    1, -4
        CRCT    1
        CMMA                a > 1 (condição while)
        DSVF    L1
        CRVL    1, 0
        CRVL    1, -4
        MULT                result * a;
        ARMZ    1, 0        result := result * a;
        CRVL    1, -4
        CRCT    1
        SUBT                a - 1;
        ARMZ    1, -4       a := a - 1;
        DSVS    L0          Desvia para o início do 'while'
L1:     NADA                fim while
        CRVL    1, 0
        IMPR                write(result);
        DMEM    1
        RTPR    1, 1
R0:     NADA                início programa
        LEIT
        ARMZ    0, 0        read(x);
        CRVL    0, 0
        CHPR    R1, 0       fat(x);
        DMEM    1
        PARA
```

Execução

```
(base) [mtuser@cp5858 mepa](main)$ ./mepa -r -i ../prova2.mepa
----------------------------------------
 Arquivo de entrada: ../prova2.mepa
 Modo Interativo:    nao
 Iniciando Execução
----------------------------------------
(entrada):10
saida: 3628800
execução finalizada
```

3 - Programa contendo a definição e chamada de uma função com passagem de ao menos um parâmetro:

Função que verifica se um determinado número é par ou impar. Isso se torna válido pois a MEPA não possui um operador 'mod', por isso, foi escrita esta função que consiste em dividir a entrada por 2, obter apenas a

parte inteira do resultado (padrão do formato 'integer'), e depois multiplicar por 2, caso o valor resultante seja menor que entrada, significa que uma unidade se perdeu, logo o resultado é 0 (ímpar), caso contrário, o resultado é 1 (par).

Código Rascal:

```
program prova3;
  var x, y: integer;

  function isPar(a: integer): integer;
  var aux: integer;
  begin
    aux := a div 2;
    aux := aux * 2;
    if aux < a then
      begin
        isPar := 0;
      end
    else
      begin
        isPar := 1;
      end
  end

begin
  read(x);
  y := isPar(x);
  write(y);
end.
```

Tabela de Simbolos

Variável	Endereço
x	0, 0
y	0, 1
isPar(...)	R1
aux	1, 0
a	1, -4
isPar	1, -5

Compilação para código MEPA:

```
        INPP
        AMEM    2         var x, y: integer;
        DSVS    R0
```

```
R1:     ENPR    1         function isPar(a: integer): integer;
        AMEM    1         var aux: integer;
        CRVL    1, -4     a
        CRCT    2
        DIVI              a div 2;
        ARMZ    1, 0      aux := a div 2;
        CRVL    1, 0
        CRCT    2
        MULT              aux * 2;
        ARMZ    1, 0      aux := aux * 2;
        CRVL    1, 0
        CRVL    1, -4
        CMME              aux < a
        DSVF    L0        desvio para o else
        CRCT    0
        ARMZ    1, -5     isPar := 0;
        DSVS    L1        desvio para fim do if
L0:     NADA              else
        CRCT    1
        ARMZ    1, -5     isPar := 1;
L1:     NADA              fim do if
        DMEM    1
        RTPR    1, 1      fim da função
R0:     NADA              início do programa
        LEIT
        ARMZ    0, 0      read(x);
        AMEM    1
        CRVL    0, 0
        CHPR    R1, 0     isPar(x);
        ARMZ    0, 1      y := isPar(x);
        CRVL    0, 1
        IMPR              write(y);
        DMEM    2
        PARA
```

Execução

```
(base) [mtuser@cp5858 mepa](main)$ ./mepa -r -i ../prova3.mepa
----------------------------------------
 Arquivo de entrada: ../prova3.mepa
 Modo Interativo:    nao
 Iniciando Execução
----------------------------------------
(entrada):99
saida: 0
execução finalizada
(base) [mtuser@cp5858 mepa](main)$ ./mepa -r -i ../prova3.mepa
----------------------------------------
 Arquivo de entrada: ../prova3.mepa
 Modo Interativo:    nao
 Iniciando Execução
----------------------------------------
```

```
(entrada):52
saida: 1
execução finalizada
```

4 - Programa contendo a definição e chamada de um procedimento ou função recursiva com passagem de ao menos um parâmetro:

Este programa imprime todos os números pares entre *n* e 0, fazendo uso de recursão e a função do exercício 3.

Código Rascal:

```
program prova4;
  var x: integer;

  function isPar(a: integer): integer;
  var aux: integer;
  begin
    aux := a div 2;
    aux := aux * 2;
    if aux < a then
      begin
        isPar := 0;
      end
    else
      begin
        isPar := 1;
      end
  end

procedure printEven(in: integer);
var cond, aux: integer;
begin
  cond := isPar(in);
  if (cond = 1) then
  begin
    write(in);
  end
  if in > 0 then
  begin
    aux := in - 1;
    printEven(aux);
  end
end

begin
    read(x);
    printEven(x);
end.
```

Tabela de Simbolos

Variável	Endereço
x	0, 0
isPar(...)	R1
aux	1, 0
a	1, -4
isPar	1, -5
printEven(...)	R2
cond	1, 0
aux	1, 1
in	1, -4

Compilação para código MEPA

```
        INPP
        AMEM    1       var x: integer;
        DSVS    R0
R1:     ENPR    1         function isPar(a: integer): integer;
        AMEM    1         var aux: integer;
        CRVL    1, -4     a
        CRCT    2
        DIVI               a div 2;
        ARMZ    1, 0      aux := a div 2;
        CRVL    1, 0
        CRCT    2
        MULT               aux * 2;
        ARMZ    1, 0      aux := aux * 2;
        CRVL    1, 0
        CRVL    1, -4
        CMME               aux < a
        DSVF    L0        desvio para o else
        CRCT    0
        ARMZ    1, -5     isPar := 0;
        DSVS    L1        desvio para fim do if
L0:     NADA              else
        CRCT    1
        ARMZ    1, -5     isPar := 1;
L1:     NADA              fim do if
        DMEM    1
        RTPR    1, 1      fim da função
R2:     ENPR    1         procedure printEven(in: integer);
        AMEM    2         var cond, aux: integer;
        AMEM    1
        CRVL    1, -4
        CHPR    R1, 1     isPar(in);
```

```
        ARMZ    1, 0      cond := isPar(in);
        CRVL    1, 0
        CRCT    1
        CMIG              cond = 1
        DSVF    L2        desvio condicional if
        CRVL    1, -4
        IMPR              write(in);
L2:     NADA              fim do if (cond = 1)
        CRVL    1, -4
        CRCT    0
        CMMA              in > 0
        DSVF    L3        desvio condicional if
        CRVL    1, -4
        CRCT    1
        SUBT              in - 1;
        ARMZ    1, 1      aux := in - 1;
        CRVL    1, 1
        CHPR    R2, 1     printEven(aux); (chamada recursiva)
L3:     NADA              fim do if in > 0
        DMEM    2
        RTPR    1, 1      fim procedure
R0:     NADA              inicio do programa
        LEIT              read(x);
        ARMZ    0, 0
        CRVL    0, 0
        CHPR    R2, 0     printEven(x);
        DMEM    1
        PARA
```

Execução:

```
(base) [mtuser@cp5858 mepa](main)$ ./mepa -r -i ../prova4.mepa
----------------------------------------
 Arquivo de entrada: ../prova4.mepa
 Modo Interativo:    nao
 Iniciando Execução
----------------------------------------
(entrada):10
saida: 10
saida: 8
saida: 6
saida: 4
saida: 2
saida: 0
execução finalizada
```