Construção de Compiladores
Período Especial
Aula 13: Funções

Bruno Müller Junior

Departamento de Informática
UFPR

2020

Introdução

- Na linguagem Pascal, quando um procedimento retorna algum valor, é chamado função.
- A concepção vem da matemática, onde $f(x)$ é uma função que recebe um parâmetro x como entrada e retorna um valor.
- Os parâmetros podem ser passados por valor ou por referência.

Sintaxe

- A regra de declaração de uma função em Pascal (regra 13) é a outra opção da regra 11, <parte de declaração de subrotinas>.
- Diferencia-se de procedimento pelo uso da palavra reservada FUNCTION e pela declaração do tipo de retorno.

```
11.  <parte de declaração de subrotinas> ::=
                <declaração de procedimento> |
                <declaração de função>
13. <declaração de função> ::= FUNCION <ident>
                               [<parâmetros formais>]
                               : <identificador>;
                               <bloco>
```

- Dentro da função, o identificador referente à função pode receber valores, como por exemplo:

```
...
function f (n:integer):integer;
begin
    ...
    f:=f(n-1)+f(n-2);
    ...
end;
```

- O símbolo `f` tem uma "vida dupla":
  - do lado esquerdo de uma atribuição, comporta-se como uma variável local (ou seja, tem um espaço de memória para armazenar valores);
  - em outras situações, comporta-se como um procedimento.

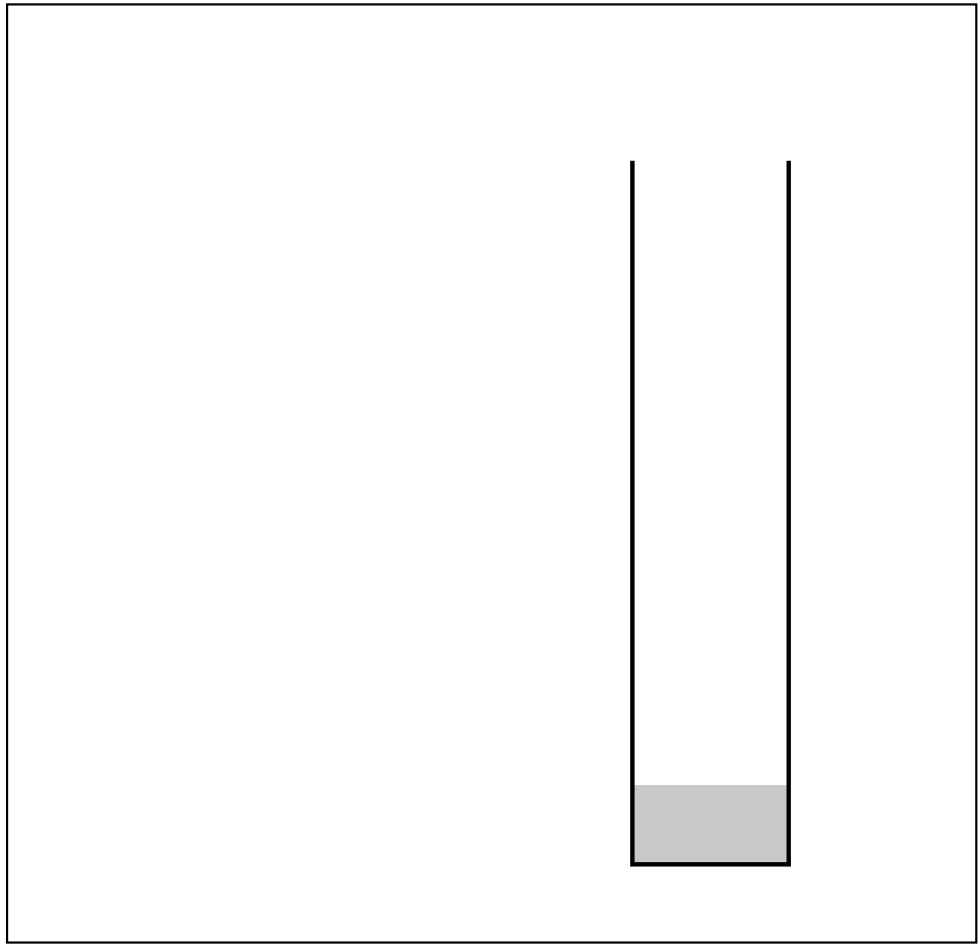
Esquema de tradução

- Para alocar o espaço para a função armazenar valores, não é necessária nenhuma nova instrução:
  - Basta abrir um espaço antes da chamada, e ao final o resultado estará no topo da pilha.
  - O endereço léxico será abaixo do primeiro parâmetro empilhado.

```
...
f(...);     { AMEM 1
              Empilha Parâmetros
              CHPR P01, k
...
```

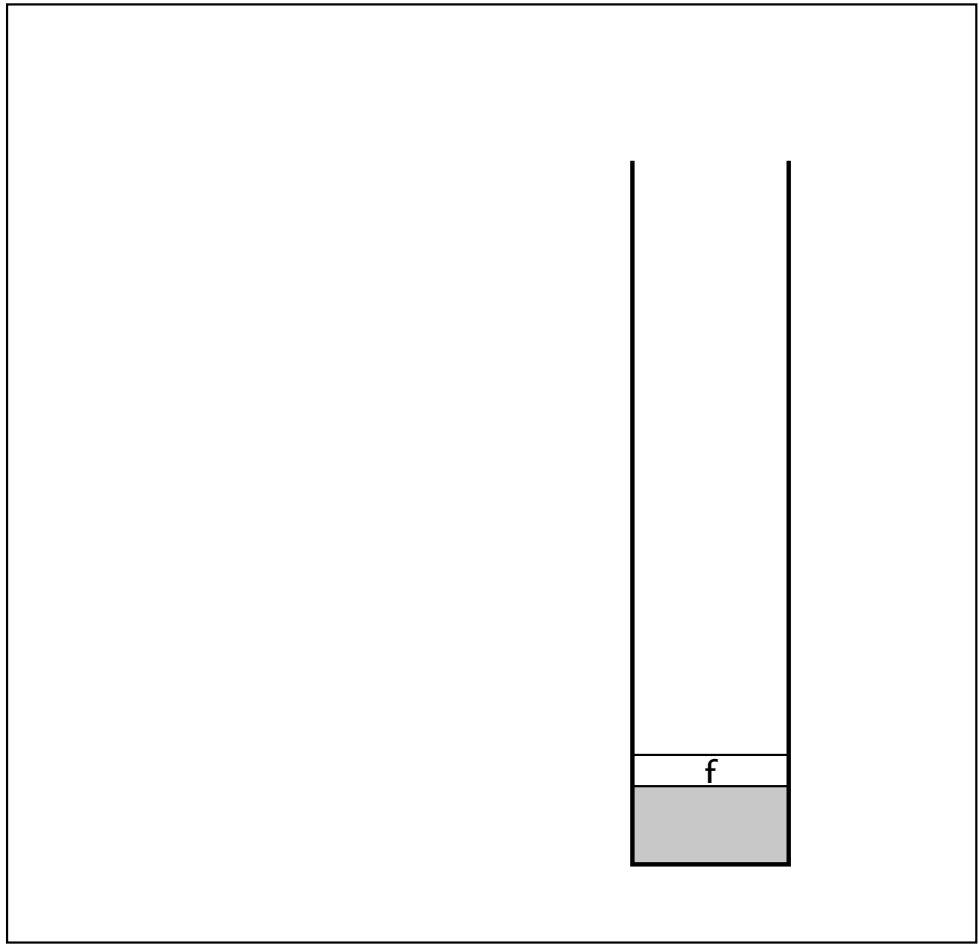
Esquema de tradução

```
x:=f(3);     ---> AMEM 1
             ---> CRCT 3
             ---> CHPR ...
             ---> ARMZ x
```



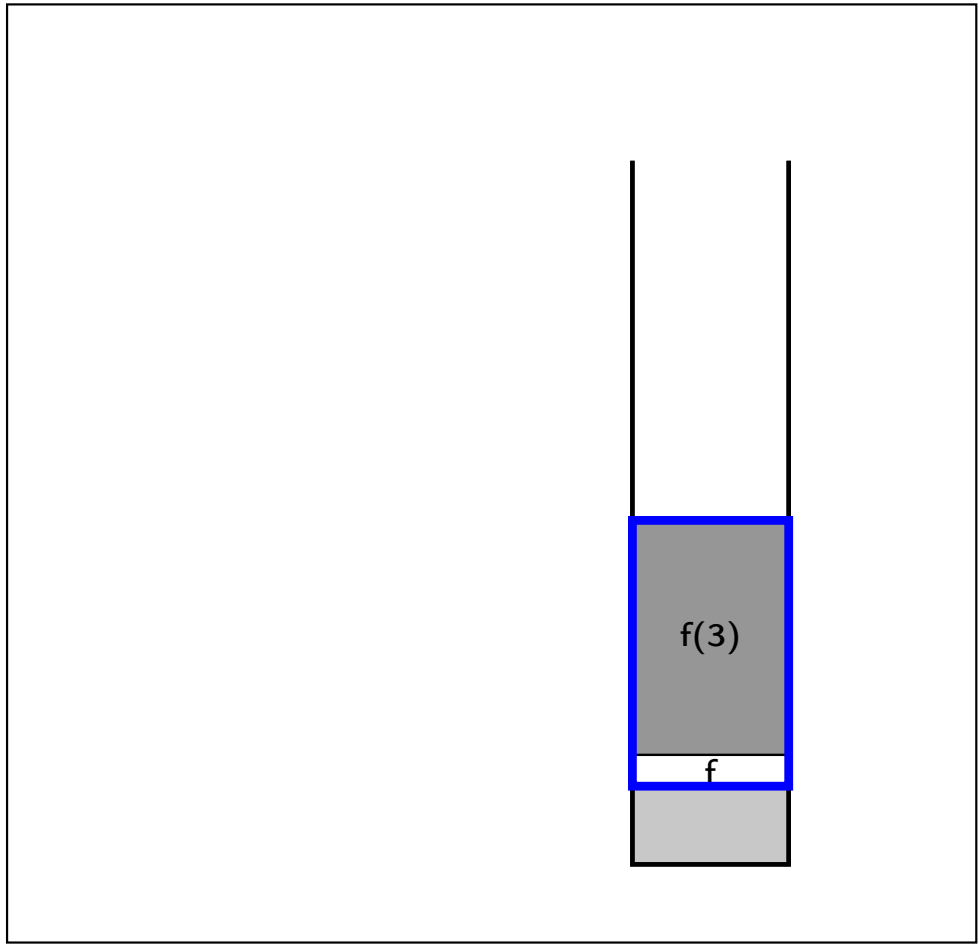
Esquema de tradução

```
x:=f(3);     ---> AMEM 1
             ---> CRCT 3
             ---> CHPR ...
             ---> ARMZ x
```



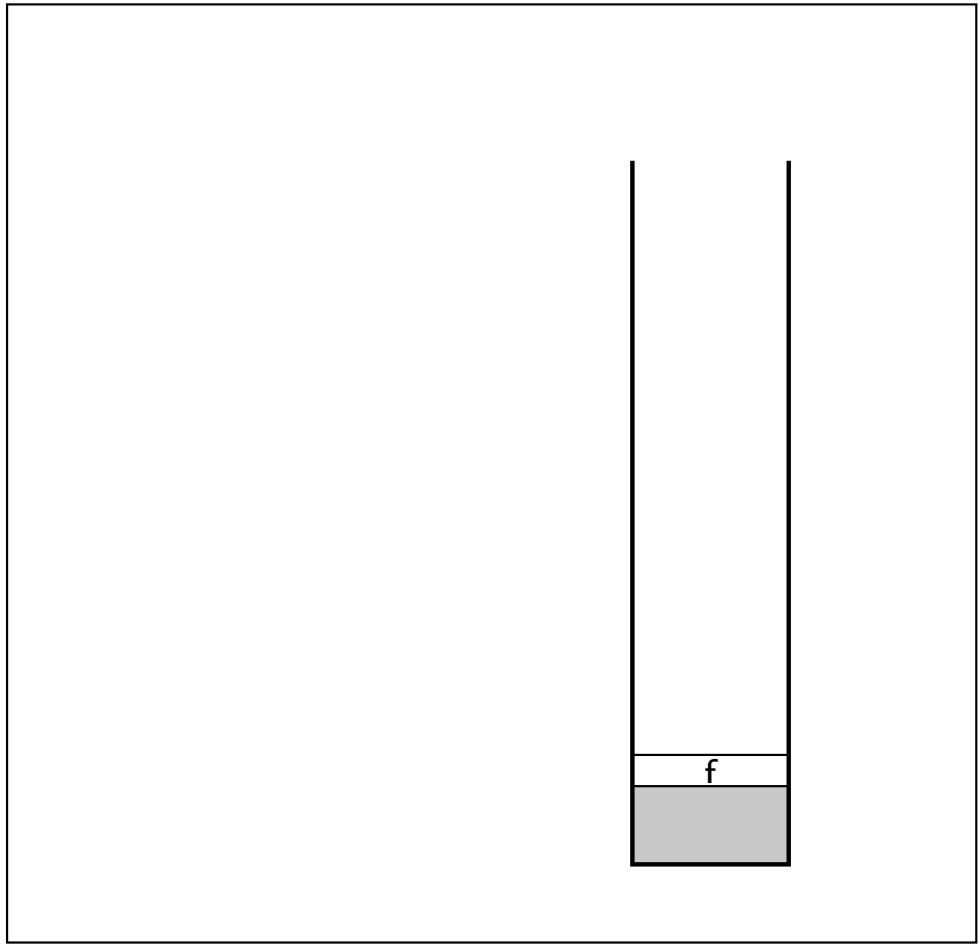
Esquema de tradução

```
x:=f(3);     ---> AMEM 1
             ---> CRCT 3
             ---> CHPR ...
             ---> ARMZ x
```



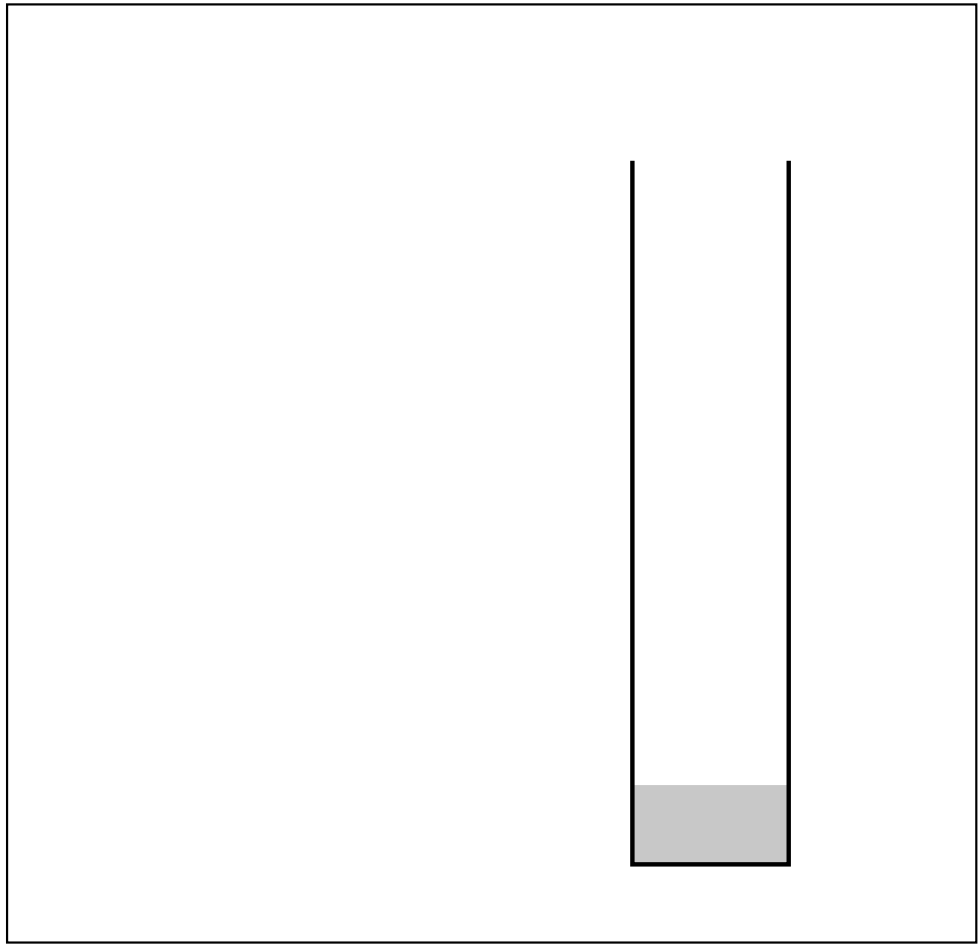
Esquema de tradução

```
x:=f(3);     ---> AMEM 1
             ---> CRCT 3
             ---> CHPR ...
             ---> ARMZ x
```



Esquema de tradução

```
x:=f(3);     ---> AMEM 1
             ---> CRCT 3
             ---> CHPR ...
             ---> ARMZ x
```



```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

Exemplo

```
program funcao (input, output);          INPP
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
INPP
AMEM 1
```

```
m  VS  0,0,int
```

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
        INPP
        AMEM 1
        DSVS R00
R01:ENPR 1
```

```
f   FUN R01,1,?,? ?{}
m   VS  0,0,int
```

Exemplo

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
        INPP
        AMEM 1
        DSVS R00
R01:ENPR 1
```

```
k  PF  1,?,int,ref
n  PF  1,?,int,vlr
f  FUN R01,1,?,? ?{}
m  VS  0,0,int
```

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
        INPP
        AMEM 1
        DSVS R00
R01:ENPR 1
```

```
k  PF  1,-4,int,ref
n  PF  1,-5,int,vlr
f  FUN R01,1,-6,int 2{[i,v][i,r]}
m  VS  0,0,int
```

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
      INPP
      AMEM 1
      DSVS R00
R01:ENPR 1
      AMEM 2
```

```
p  VS  1,1,int
q  VS  1,0,int
k  PF  1,-4,int,ref
n  PF  1,-5,int,vlr
f  FUN R01,1,-6,int 2{[i,v][i,r]}
m  VS  0,0,int
```

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
      INPP
      AMEM 1
      DSVS R00
R01:ENPR 1
      AMEM 2
      CRVL 1,-5
      CRCT 2
      CMME
      DSVF R02
      CRVL 1,-5
      ARMZ 1,-6
      CRCT 0
      ARMI 1,-4
      DSVS R03
```

```
p  VS  1,1,int
q  VS  1,0,int
k  PF  1,-4,int,ref
n  PF  1,-5,int,vlr
f  FUN R01,1,-6,int 2{[i,v][i,r]}
m  VS  0,0,int
```

Exemplo

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.

Verifica se a chamada da função tem
2 parâmetros:
   - o segundo tem de ser uma variável!
```

```
      INPP
      AMEM 1
      DSVS R00
R01:ENPR 1
      AMEM 2
      CRVL 1,-5
      CRCT 2
      CMME
      DSVF R02
      CRVL 1,-5
      ARMZ 1,-6
      CRCT 0
      ARMI 1,-4
      DSVS R03
R02:NADA
      AMEM 1
      CRVL 1,-5
      CRCT 1
      SUBT
      CREN 1,0
      CHPR R01,1
      AMEM 1
      CRVL 1,-5
      CRCT 2
      SUBT
      CREN 1,1
      CHPR R01,1
      SOMA
      ARMZ 1,-6
      CRVL 1,0
      CRVL 1,1
      SOMA
      CRCT 1
      SOMA
      ARMI 1,-4
R03:NADA
```

```
p  VS  1,1,int
q  VS  1,0,int
k  PF  1,-4,int,ref
n  PF  1,-5,int,vlr
f  FUN R01,1,-6,int 2{[i,v][i,r]}
m  VS  0,0,int
```

Exemplo

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
     INPP
     AMEM 1
     DSVS R00
R01:ENPR 1
     AMEM 2
     CRVL 1,-5
     CRCT 2
     CMME
     DSVF R02
     CRVL 1,-5
     ARMZ 1,-6
     CRCT 0
     ARMI 1,-4
     DSVS R03
R02:NADA
     AMEM 1
     CRVL 1,-5
     CRCT 1
     SUBT
     CREN 1,0
     CHPR R01,1
     AMEM 1
     CRVL 1,-5
     CRCT 2
     SUBT
     CREN 1,1
     CHPR R01,1
     SOMA
     ARMZ 1,-6
     CRVL 1,0
     CRVL 1,1
     SOMA
     CRCT 1
     SOMA
     ARMI 1,-4
R03:NADA
     CRVL 1,-5
     IMPR
     CRVI 1,-4
     IMPR
```

```
p  VS  1,1,int
q  VS  1,0,int
k  PF  1,-4,int,ref
n  PF  1,-5,int,vlr
f  FUN R01,1,-6,int 2{[i,v][i,r]}
m  VS  0,0,int
```

Exemplo

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
     INPP
     AMEM 1
     DSVS R00
R01:ENPR 1
     AMEM 2
     CRVL 1,-5
     CRCT 2
     CMME
     DSVF R02
     CRVL 1,-5
     ARMZ 1,-6
     CRCT 0
     ARMI 1,-4
     DSVS R03
R02:NADA
     AMEM 1
     CRVL 1,-5
     CRCT 1
     SUBT
     CREN 1,0
     CHPR R01,1
     AMEM 1
     CRVL 1,-5
     CRCT 2
     SUBT
     CREN 1,1
     CHPR R01,1
     SOMA
     ARMZ 1,-6
     CRVL 1,0
     CRVL 1,1
     SOMA
     CRCT 1
     SOMA
     ARMI 1,-4
R03:NADA
     CRVL 1,-5
     IMPR
     CRVI 1,-4
     IMPR
     DMEM 2
     RTPR 1,2
```

~~p VS 1,1,int~~
~~q VS 1,0,int~~
~~k PF 1,-4,int,ref~~
~~n PF 1,-5,int,vlr~~
f FUN R01,1,-6,int 2{[i,v][i,r]}
m VS 0,0,int

Exemplo

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
     INPP
     AMEM 1
     DSVS R00
R01:ENPR 1
     AMEM 2
     CRVL 1,-5
     CRCT 2
     CMME
     DSVF R02
     CRVL 1,-5
     ARMZ 1,-6
     CRCT 0
     ARMI 1,-4
     DSVS R03
R02:NADA
     AMEM 1
     CRVL 1,-5
     CRCT 1
     SUBT
     CREN 1,0
     CHPR R01,1
     AMEM 1
     CRVL 1,-5
     CRCT 2
     SUBT
     CREN 1,1
     CHPR R01,1
     SOMA
     ARMZ 1,-6
     CRVL 1,0
     CRVL 1,1
     SOMA
     CRCT 1
     SOMA
     ARMI 1,-4
R03:NADA
     CRVL 1,-5
     IMPR
     CRVI 1,-4
     IMPR
     DMEM 2
     RTPR 1,2
R00:NADA
     AMEM 1
     CRCT 3
     CREN 0,0
     CHPR R01,0
     IMPR
     CRVL 0,0
     IMPR
```

```
f  FUN R01,1,-6,int 2{[i,v][i,r]}
m  VS  0,0,int
```

Exemplo

```
program funcao (input, output);
var  m: integer;
  function f(n:integer;
         var k:integer):integer;
  var p, q: integer;
  begin
    if n<2 then
      begin
        f:=n; k:=0
      end
    else
      begin
        f:=f(n-1,p) + f(n-2,q);
        k:=p+q+1
      end;
    write(n,k);
  end;
begin
  write(f(3,m),m);
end.
```

```
     INPP
     AMEM 1
     DSVS R00
R01:ENPR 1
     AMEM 2
     CRVL 1,-5
     CRCT 2
     CMME
     DSVF R02
     CRVL 1,-5
     ARMZ 1,-6
     CRCT 0
     ARMI 1,-4
     DSVS R03
R02:NADA
     AMEM 1
     CRVL 1,-5
     CRCT 1
     SUBT
     CREN 1,0
     CHPR R01,1
     AMEM 1
     CRVL 1,-5
     CRCT 2
     SUBT
     CREN 1,1
     CHPR R01,1
     SOMA
     ARMZ 1,-6
     CRVL 1,0
     CRVL 1,1
     SOMA
     CRCT 1
     SOMA
     ARMI 1,-4
R03:NADA
     CRVL 1,-5
     IMPR
     CRVI 1,-4
     IMPR
     DMEM 2
     RTPR 1,2
R00:NADA
     AMEM 1
     CRCT 3
     CREN 0,0
     CHPR R01,0
     IMPR
     CRVL 0,0
     IMPR
     DMEM 1
     PARA
```

~~f FUN R01,1,-6,int 2{[i,v][i,r]}~~
~~m VS 0,0,int~~

Execução

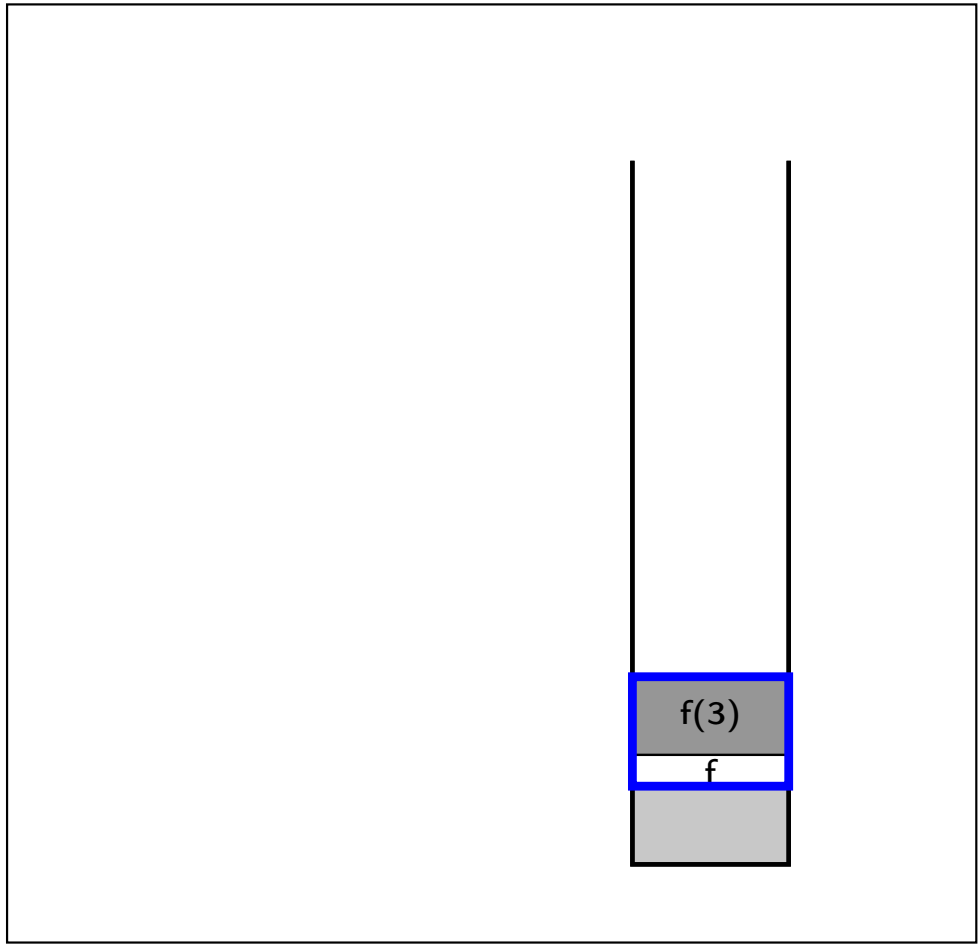
- Como não há novidades com relação à execução, não será apresentada uma execução detalhado do programa traduzido.
- Como não há novidades com relação à execução, não será apresentada uma execução detalhado do programa traduzido.
- Exercício: simule a execução na MEPA

Execução

- Como não há novidades com relação à execução, não será apresentada uma execução detalhado do programa traduzido.
- Como não há novidades com relação à execução, não será apresentada uma execução detalhado do programa traduzido.
- Exercício: simule a execução na MEPA
- Será apresentado o funcionamento do trecho de código abaixo:

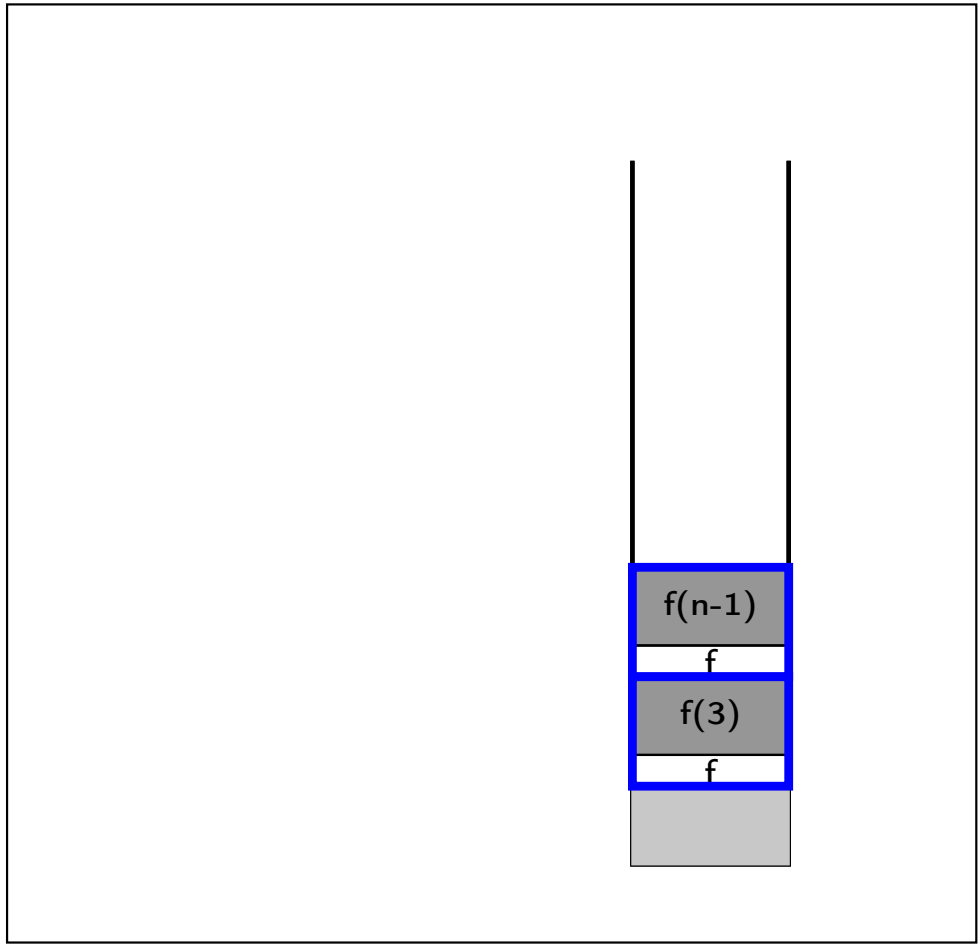
```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```

```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```

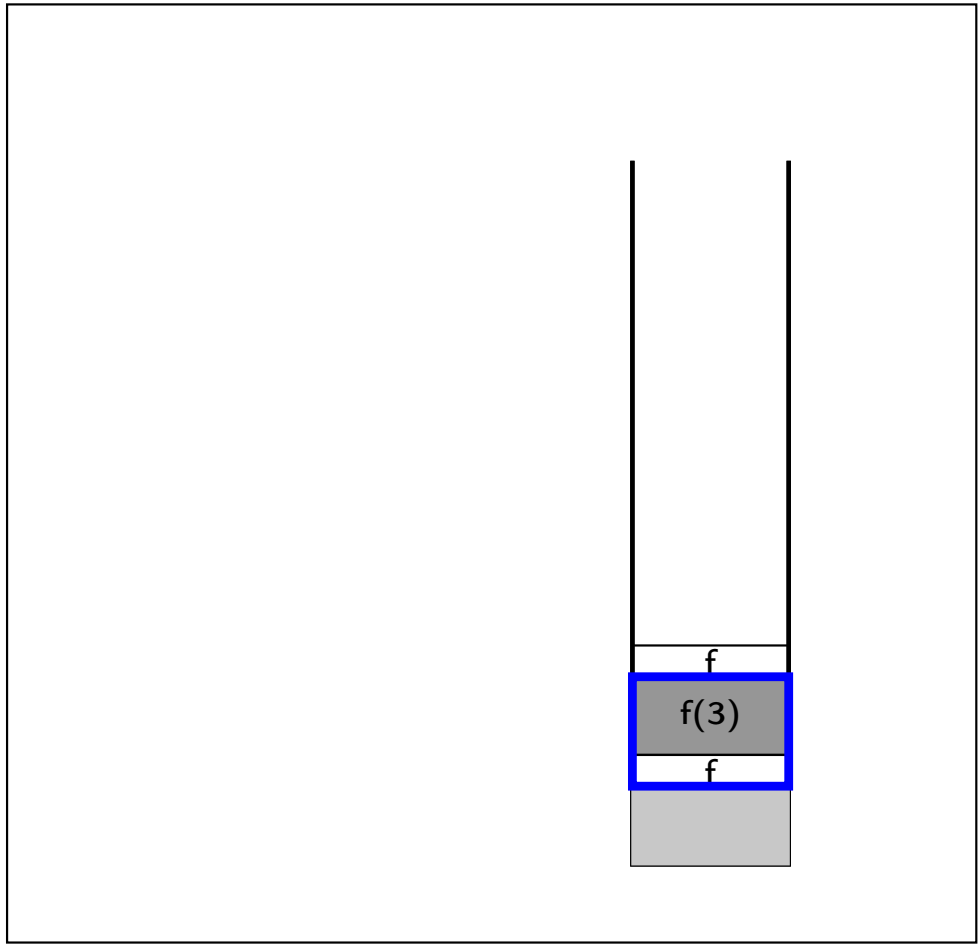


Execução

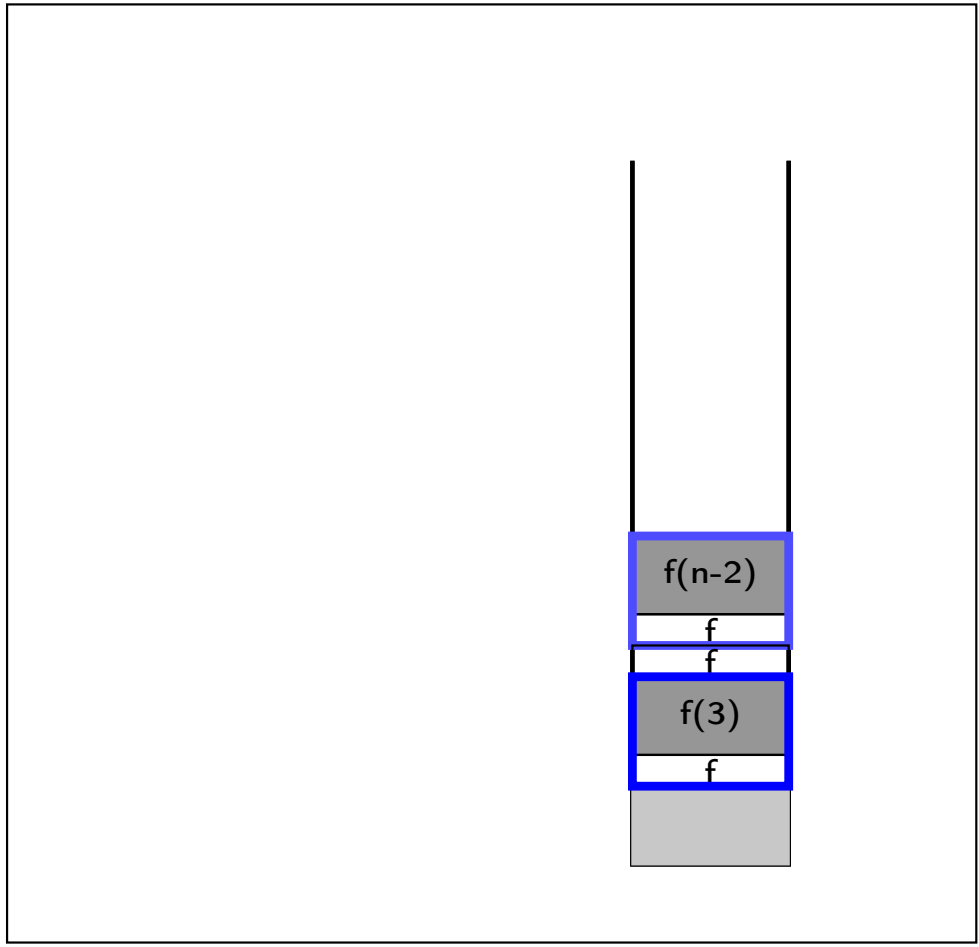
```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```



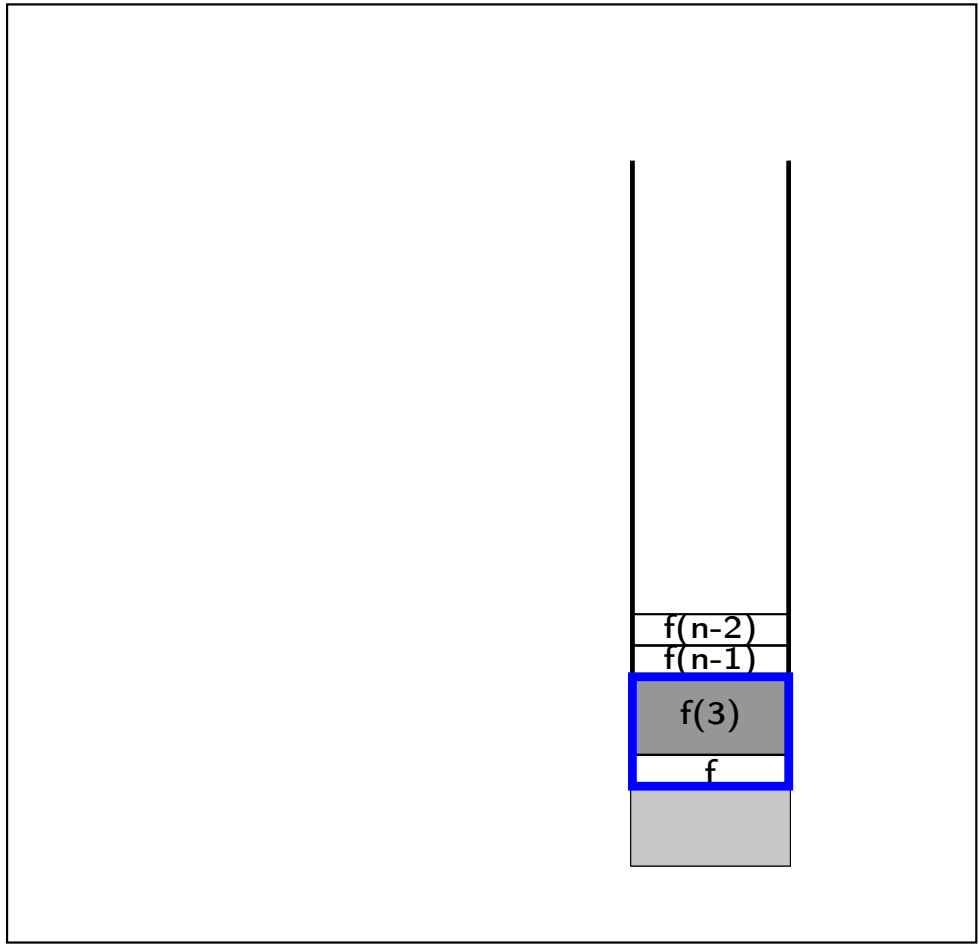
```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```



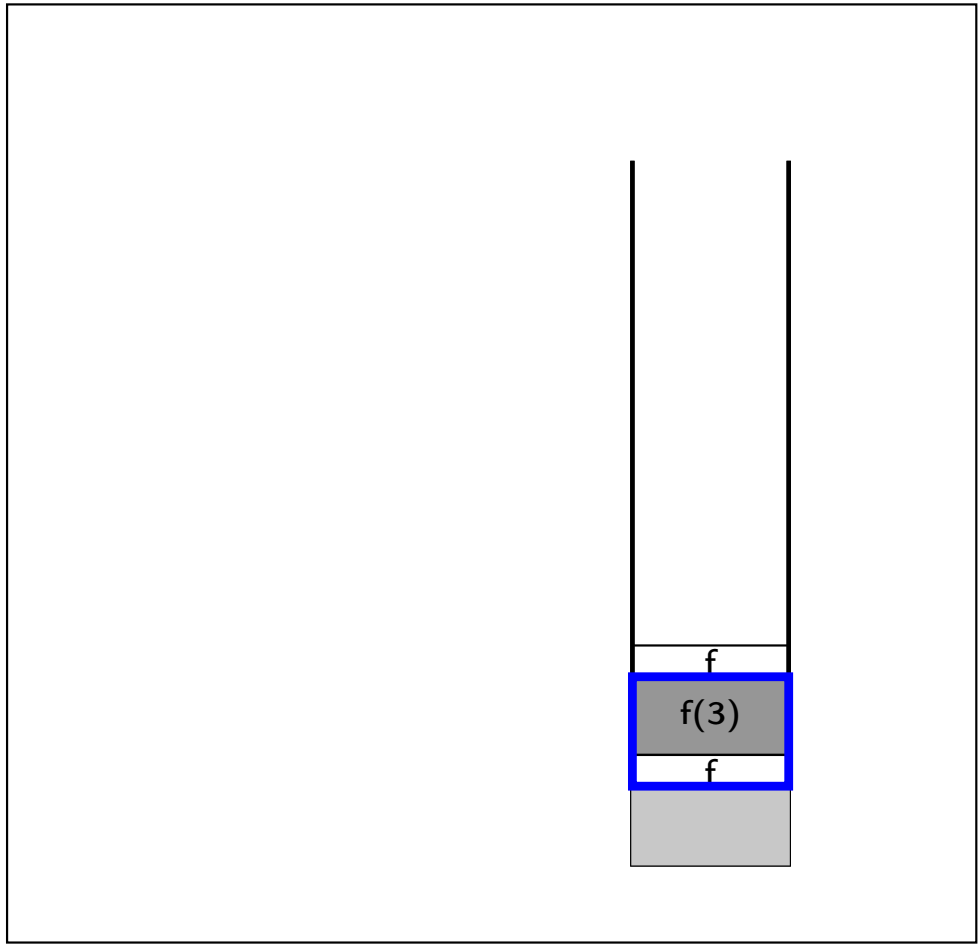
```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```



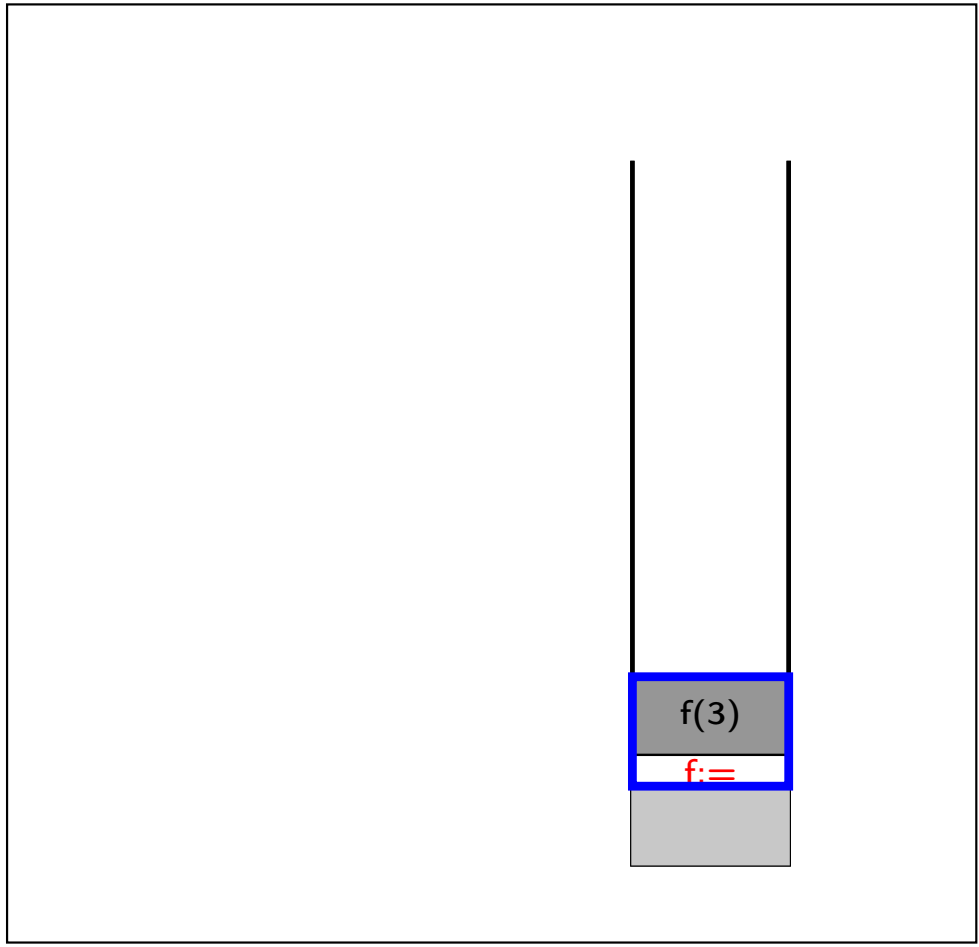
```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```



```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```



```
...
   function f (n:integer):integer;
   begin
      ...
      f:=f(n-1)+f(n-2);
      ...
   end;
...
 x:=f(3);
...
```



Projeto

- Inclua funções no compilador. Algumas observações:
  - Na tabela de símbolos, a categoria função deve conter tanto as informações de variável (como tipo e end. léxico) quanto de procedimento (rótulo de entrada, parâmetros).
  - O endereço léxico da função é $k, (-4 + n)$, onde n é o número de parâmetros.

- Página para anotações

Licença

- Slides desenvolvidos somente com software livre:
  - LaTeX usando beamer;
  - Inkscape.
- Licença:
  - Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License. `http://creativecommons.org/licenses/by-nc-nd/2.5/br/`