

# **PRÁCTICA GENERACIÓN DE CÓDIGO**

Inicialización

Construcción

With

Métodos

# Generación de código de declaración de multiples variables

## Fuente

```
procedure main()
begin
  Var a,b,c:Integer=10;
  print a,b,c;
end
```

## Código generado

```
0000      ICall  main
0005      IExit
0006 main:  ILink 12

0011      IPushLit  4 10

0020      IStoreBVar  4 -4
0029      IStoreBVar  4 -8
0038      IPopBVar   4 -12

0047      IPushBVar  4 -4
0056      IPrintInt
0057      IPushBVar  4 -8
0066      IPrintInt
0067      IPushBVar  4 -12
0076      IPrintInt
0077      IUnlink
0078      IRet
```

## Generación de código de construcción de array

### Fuente

```
procedure main()
begin
  Var a: array [3] of Integer={1,2,3};
end
```

### Código generado

```
0000      ICall  main
0005      IExit

0006 main:  ILink 12

0011      IPushLit  4 3
0020      IPushLit  4 2
0029      IPushLit  4 1
0038      IPopBVar   12 -12

0047      IUnlink
0048      IRet
```

## Sintaxis de la instrucción with

Rule <Instruccio> ::= with <IdFactor> do

{ <DecFun> | <DecProc> | <DecVar> | <DecTipus> | <instruccio> }  
"end"

Rule <IdFactor> ::= Identificador

( "(" [<Expressio> {, <Expressio> } ] ")" | \$ )

<Acces>

( = <Expressio> | \$ )

Rule <Acces> ::=

{

"[" <Expressio> {, <Expressio> } ]" |

. identificador ( \$ | "(" [ <Expressio> {, <Expressio> } ] ")" ) |

^

}

## Generación de código de with

### Fuente

```
Type rec=record
  camp1,camp2,camp3:Integer;
end;
```

```
Procedure main()
```

```
begin
```

```
  Var a:rec;
```

```
  with a do
```

```
    camp1=10;
```

```
    camp2=20;
```

```
    camp3=30;
```

```
  end
```

```
end
```

### Código generado

```
0000      ICall  main
0005      IExit
0006 main:  ILink 16

0011      IPushAddressBVar  -12
0016      IPopBVar  4 -16

0025      IPushLit  4 10
0034      IPopDispVar  4 -16 0

0047      IPushLit  4 20
0056      IPopDispVar  4 -16 4

0069      IPushLit  4 30
0078      IPopDispVar  4 -16 8

0091      IUnlink
0092      IRet
```

## Consideraciones sobre la generación de código de with

- La expresión de la instrucción with ha de generar la dirección del record.
- La dirección del record se ha de guardar en una variable local para poderla utilizar para acceder a los campos.
- A los campos del record del with se acceden como
  - Dirección record+ desplazamiento campo
  - La dirección de un campo será
    - Variable local accedida por B+desp  
\*(B+desp. dirección record)+desp. campo
    - Variable local accedida por display  
\*(\*(B+nivel)+desp. dirección record)+desp. campo

## Sintaxis Factor

Rule <factor> ::= ! <factor> | - <factor> | "(" <Expressio> ")" |  
Numero | Caracter | String |  
"{" <Expressio> {, <Expressio> } }" |  
<IdFactor> |  
"new" <tipus> | "Null"

Rule <IdFactor> ::= Identificador  
( "(" [<Expressio> {, <Expressio> } ] ")" | \$ )  
<Acces>  
( = <Expressio> | \$ )

Rule <Acces> ::=  
{  
  "[" <Expressio> {,<Expressio> }]" |  
  . identificador ( \$ | "(" [ <Expressio> {, <Expressio> } ] ")" ) |  
  ^  
}

## Acceso a dato permanente

### Fuente

```
type rec=record
  camp1,camp2,camp3:Integer;
end;

procedure main()
begin
  Var a:rec;
  a.camp2=10;
  print a.camp2;
end
```

### Código generado

```
0000      ICall  main
0005      IExit
0006 main:  ILink 12

0011      IPushLit  4 10
0020      IPopBVar  4 -8

0029      IPushBVar  4 -8
0038      IPrintInt

0039      IUnlink
0040      IRet
```

## Lectura de dato temporal en pila

### Fuente

```
type rec=record
  camp1,camp2,camp3:Integer;
end;
function f():rec
Begin
  var a:rec;
  a.camp2=10;
  return a;
end
procedure main()
Begin
  print f().camp2;
end
```

### Código generado

```
0051 main:   ILink 0
0056        IAddSP  -12
0061        ICall  f
0066        IAddSP  0
0071        IPushSP
0072        IPushLit  4 4
0081        IAddInt
0082        IPushInd  4
0087        IMoveTop  4 12
0096        IPrintInt
0097        IUnlink
0098        IRet
```

## Lectura de dato temporal en variable local

### Fuente

```
type rec=record
  camp1,camp2,camp3:Integer;
end;
function f():rec
Begin
  var a:rec;
  a.camp2=10;
  return a;
end
procedure main()
Begin
  print f().camp2;
end
```

### Código generado

```
0051 main:   ILink 12
0056        IAddSP  -12
0061        ICall  f
0066        IAddSP  0
0071        IPopBVar  12 -12
0080        IPushBVar  4 -8
0089        IPrintInt
0090        IUnlink
0091        IRet
```

## Llamar a un método

### Fuente

```
type rec=record
  camp1,camp2:Integer;
end;

function this:rec.m(x:integer):integer
Begin
  return this.camp1+this.camp2;
End

procedure main()
Begin
  var a:array [3] of rec;
  print a[1].m(128);
end
```

### Código generado

```
0051 main:    ILink 24

0056         IAddSP  -4
0061         IPushLit  4 128

0070         IPushAddressBVar -24
0075         IPushLit  4 1
0084         IPushLit  4 8
0093         IMultInt
0094         IAddInt

0095         ICall  rec::m
0100         IAddSP  8

0105         IPrintInt
0106         IUnlink
0107         IRet
```

## Llamar a un método con objeto temporal en pila

### Fuente

```
type rec=record
  camp1,camp2:Integer;
end;
function f():rec
begin
  return rec(10,20);
end
function this:rec.m(x:integer):integer
begin
  return this.camp1+this.camp2;
end
procedure main()
begin
  print f().m(128);
end
```

### Código generado

```
0087 main:   ILink 0
0092        IAddSP  -8
0097        ICall  f
0102        IAddSP  0
0107        IAddSP  -4
0112        IPushLit  4 128
0121        IPushAddressSPVar 8
0126        ICall  rec::m
0131        IAddSP  8
0136        IMoveTop  4 8
0145        IPrintInt
0146        IUnlink
0147        IRet
```

## Llamar a un método con objeto temporal en variable

### Fuente

```
type rec=record
  camp1,camp2:Integer;
end;
function f():rec
begin
  return rec(10,20);
end
function this:rec.m(x:integer):integer
begin
  return this.camp1+this.camp2;
end
procedure main()
begin
  print f().m(128);
end
```

### Código generado

```
0087 main:   ILink 8

0092        IAddSP  -8
0097        ICall  f
0102        IAddSP  0
0107        IPopBVar  8 -8

0116        IAddSP  -4

0121        IPushLit  4 128

0130        IPushAddressBVar  -8

0135        ICall  rec::m
0140        IAddSP  8

0145        IPrintInt
0146        IUnlink
0147        IRet
```