Conceptos y Paradigmas de Lenguajes de Programación
2024

Práctica Nro. 5
Pilas de ejecución

Objetivo: Interpretar cómo se organiza la memoria de datos durante la ejecución de un programa con llamados a subrutinas.

Ejercicio 1: Explique claramente cual es la utilidad del registro de activación y que representan cada una de sus partes.(Basado en el modelo debajo detallado)

Modelo de registro de activación

Head (prog principal)
Pto retorno
EE (enlace estático)
ED (enlace dinámico)
Variables...
...
Parámetros ...
....
Procedimientos ....
....
Funciones ...
....
Valor de retorno

Ejercicio 2: Dado el siguiente programa escrito en Pascal-like, continuar la realización de las pilas de ejecución hasta finalizar las mismas.

a) Siguiendo la cadena estática b) Siguiendo la cadena dinámica

```
Program Main
        Var a: array[1..10] of integer;
             x,y,z:integer
        Procedure A ()
                var y,t: integer;
                begin
                   a(1):= a(1)+1;z:=z+1;
                   t:=1; y:=2;
                   B(); a(y):=a(y)+3; y:=y+1;
                   If z=11 Then Begin
                         a(z-1):=a(z-2)      +      3;
                z:=z-4;
                         a(z-y):=a(z) – a(y) + 5;
                    End;
end;
Function t():integer
        begin
        y:=y+1; z:=z-6;
        return(y+x);
        end;
```

```
Procedure B()
        var d:integer;
        Procedure I ()
        begin
                x:=0; x:=x+6;
        end;
        begin
                x:=x+t; d:=0;
                while x>d do begin
                        I(); x:=x-1;
                        d:=d + 2;
                end;
        end;
begin
        For x:=1 To 10 do a(x):=x;
        x:=5; y:=1; z:=10;
        A();
        For x:=1 To 10 do write(a(x),x);
end.
```

Nota: La forma de evaluación de este lenguaje es de izquierda a derecha

Siguiendo la cadena estática

	*** Reg Activ Main
*1	Pto retorno
	A(1)= ~~1~~
	A(2)= 2
	A(3)= 3
	A(4)= 4
	A(5)= 5
	A(6)= 6
	A(7)= 7
	A(8)= 8
	A(9)= 9
	A(10)= 10
	X= ~~1..10~~ 5
	Y= ~~1~~ – 2
	Z=~~10~~ – ~~11~~ – 5
	Procedure A
	Function T
	Procedure B
	VR . . . . .
*2	***Reg Activ A
	Pto Retorno
	EE (*1)
	ED (*1)
	Y = 2
	T = 1
	VR . . . . .
	*** Reg Activ B
	Pto Retorno
	EE
	ED
	D =
	Procedure I
	VR . . ¿? . .
	*** Reg Activ…(a partir de acá lo debe continuar…
	…….

Siguiendo la cadena dinámica

	*** Reg Activ Main
*1	Pto retorno
	A(1)= ~~1, 2,~~ 5
	A(2)= 2
	A(3)= 3
	A(4)= 4
	A(5)= 5
	A(6)= 6
	A(7)= 7
	A(8)= 8
	A(9)= 9
	A(10)= 10
	X= ~~1..10~~ 5
	Y= ~~1~~ – 2
	Z=~~10~~ – 11
	Procedure A
	Function T
	Procedure B
	VR . . . . .
*2	***Reg Activ A
	Pto Retorno
	EE (*1)
	ED (*1)
	Y = 2
	T = 1
	VR . . . . .
*3	*** Reg Activ B
	Pto Retorno
	EE (*1)
	ED (*2)
	D =
	Procedure I
	VR . . ¿? . .
*4	*** Reg Activ…(a partir de acá lo debe continuar…
	…….

Conceptos y Paradigmas de Lenguajes de Programación
2024

Ejercicio 3: Sea el siguiente programa escrito en Pascal-like. Realice la pila de ejecución
a) Siguiendo la cadena estática
b) Siguiendo la cadena dinámica

```
PROGRAM P1;
var
        a:integer;
        b:char;
        c: array[1..10] of integer

Procedure PP1;
var
        a:char;
        p:integer;
Function x: integer;
var
        z:integer;
begin
        a:="j";
        z=-1;
        return z;
end;

Begin
        p:=x;
        write(a);
        p:=x+3;
        c[p]=8;
        p:=x+2;
        c[p]=x;
end;
```

```
Procedure x;
var
        b:char;
Procedure PP2;
Begin
        write("para qué estoy aquí?");
end;
Begin
        a:=1;
        c[a]:=4;
        b:="a";
        write(concat(c[1],b)); /*concat convierte a string los
parámetros, concatena y retorna un string;*/
        PP1();
        b:="b";
        write(concat(c[5],b)); /*concat convierte a string los
parámetros, concatena y retorna un string;*/
End;
BEGIN
a:=3;
b:="c";
for a:=3 to 10 do
        begin
                c[a]:=2*a;
        end;
x;
write(b);
write(a);
for a:=1 to 10 do
        write(c[a]-3);
END.
```

Nota: La forma de evaluación de este lenguaje es de izquierda a derecha

Conceptos y Paradigmas de Lenguajes de Programación
2024

Ejercicio 4: Sea el siguiente programa escrito en Pascal-like. Realice la pila de ejecución
a) Siguiendo la cadena estática
b) Siguiendo la cadena dinámica

```
Procedure Main;
        var x, y: integer;
        vec: array[1..7] of integer;
        Function B:integer;
                var y:integer;
                begin
                   y:=4; x:= y - 2;
                   return (x);
                end;
        Procedure D;
                var i, x: integer;
                vec: array[1..7] of integer;
                Procedure A;
                        var y:integer;
                        begin
                             y:=x + 5; vec(i + 2):=
                vec(i + 2) + y;
                           x:= x +B; C;
                        end;
                Function B:integer;
                        begin
                           vec(i):= y + 2; i:=i+2;
                           vec(i):= vec(1) * i;
                           return ( vec(i)-vec(1) );
                        end;
                begin
                   for x:= 1 to 7 do vec(x):= 1;
                   x:=1; i:= 2;
                   if y = 7 then A; else C;
                   for x:= 1 to 7 do write(vec(x));
                end;
```

```
        Procedure C;
                var i, y: integer;
                begin
                        i:= 1; y:= 6; x:= x + B;
                        vec(2):= vec(2) * x;
                        while (i < y) do begin
                            vec(i):= vec(i) + B - 1;
                            i:= i + 3;
                        end;
                        y:= y - 4;
                end;
        begin
           for x:= 1 to 7 do vec(x):= x;
           x:= 3; y:= B+5; D;
           if (x = 2) then begin
                vec(x):= vec(x) + 2;
                vec(x + 3):= vec(x) * 3;
            end;
          for x:= 1 to 7 do write(vec(x));
        end.
```

Nota: La forma de evaluación de este lenguaje es de izquierda a derecha

Conceptos y Paradigmas de Lenguajes de Programación
2024

Ejercicio 5: Sea el siguiente programa escrito en Pascal-like. Realice la pila de ejecución
a) Siguiendo la cadena estática
b) Siguiendo la cadena dinámica
c) La sentencia x:= c + 5 +x, podría reemplazarse por x:= x + c + 5? Justifique la respuesta

```
Program Main;
        Var x, y, z:integer;
        a, b: array[1..6] of integer;
        Procedure B;
                var y,x: integer;
                Procedure C;
                   var c:integer;
                   begin
                         y:= y + 2; c:=2;
                         a(x):=a(x)*y;
                         if (y >7) then
                                  b(y-6)=b(4)*2+b(y
                         -6);
                         D;
                   end;
                begin
                   x:=2; y:= x + 3;
                   C; x:= x + 1; write (x,y);
                End;
        Procedure D;
                begin
                    x:= c + 5 + x;
                    y:= y + 2;
                end;
```

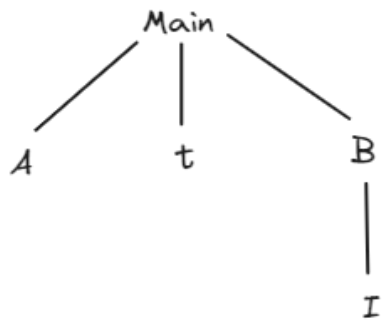
```
        Function C: integer;
                begin
                   b(x):= b(x) + 1;
                    x:= x + 1;
                    a(y):=a(y)+b(x)+3;
                    a(x+2)=a(x) + 2;
                    return b(x);
                end
begin
     x:= 1; Y:= 2;
     for z:=1 to 6 do begin
           a(z):= z;
           b(z):= z + 2;
      end;
     B;
     for z:= to 6 do write (a(z), b(z));
end.
```

Nota:La forma de evaluación de este lenguaje es de izquierda a derecha

Ejercicio 1

Registro de Activación	
Es una estructura de datos fundamental en la ejecución de programas, especialmente en entornos donde se utilizan procedimientos o funciones. Su utilidad principal radica en mantener la información necesaria para la ejecución de un procedimiento o función específica, permitiendo la gestión eficiente de la pila de llamadas y la correcta manipulación de variables locales, parámetros y contexto de ejecución.	
Partes del Registro	***Explicación***
Head (programa principal)	Es la parte inicial del registro de activación y generalmente contiene la siguiente información: **current** (dirección base del registro de activación de la unidad que se esté ejecutando actualmente) y **free** (próxima dirección libre en la pila)
Punto de Retorno	Cuando una rutina llama a otra y esta última termina, el **punto de retorno** es la dirección de memoria donde continúa la ejecución.
Enlace Estático	Puntero a la dirección base del registro de activación de la rutina que estáticamente la contiene.
Enlace Dinámico	Puntero a la dirección base del registro de activación de la rutina llamadora.
Variables	**Variables** que conforman la unidad, que van reemplazando sus valores de acuerdo a la ejecución del programa.
Parámetros	Contiene los valores de los **parámetros** pasados al **procedimiento** o **función** en el momento de la llamada. Estos pueden ser tanto valores como referencias a objetos, dependiendo del lenguaje de programación.
Procedimientos	**Procedimientos** definidos dentro de la unidad (identificadores)
Funciones	**Funciones** definidas dentro de la unidad (identificadores)
Valor de Retorno	**Valores retornados** por las **funciones** que se llamen dentro de la unidad, ya que una vez que estas finalizan, sus **Registros de Activación** se desalocan, y la unidad llamante debe almacenar esos **valores.**

Ejercicio 2



Código	Cadena estática	Cadena dinámica

Program Main
Var a: array[1..10] of integer;
x,y,z:integer
Procedure A ()
var y,t: integer;
begin
a(1):= a(1)+1;z:=z+1;
t:=1; y:=2;
B(); a(y):=a(y)+3; y:=y+1;
If z=11 Then Begin
a(z-1):=a(z-2) + 3;
z:=z-4;
a(z-y):=a(z) – a(y) + 5;
End;
end;
Function t():integer
begin
y:=y+1; z:=z-6;
return(y+x);
end;
Procedure B()
var d:integer;
Procedure I ()
begin
x:=0; x:=x+6;
end;
begin
x:=x+t; d:=0;
	*1 ***Registro de Activación Main
	Punto de Retorno
	a(1) = 1 -> 2
	a(2) = 2 -> 5
	a(3) = 3
	a(4) = 4
	a(5) = 5
	a(6) = 6
	a(7) = 7
	a(8) = 8
	a(9) = 9
	a(10) = 10
	x = 1...10 -> 5 -> 12 -> 0 -> 6 -> 5 -> 0 -> 6 -> 5 -> 0 -> 6 -> 5 -> 1...10
	y = 1 -> 2
	z = 10 -> 11 -> 5
	procedure A
	function t
	procedure B
	Valor de Retorno
	
	*2 ***Registro de Activación A
	Punto de Retorno
	Enlace Estático (*1)
	Enlace Dinámico (*1)
	t = 1
	y = 2 -> 3
	Valor de Retorno
	
	*3 ***Registro de Activación B
	Punto de Retorno
	Enlace Estático (*1)
	Enlace Dinámico (*2)
	d = 0 -> 2 -> 4 -> 6
	Procedure I
	Valor de Retorno 7
		*1 ***Registro de Activación Main
		Punto de Retorno
		a(1) = 1 -> 2
		a(2) = 2 -> 5
		a(3) = 3
		a(4) = 4 -> 9
		a(5) = 5
		a(6) = 6
		a(7) = 7
		a(8) = 8
		a(9) = 9
		a(10) = 10 -> 12
		x = 1...10 -> 5 -> 6 -> 0 -> 6 -> 5 -> 0 -> 6 -> 5 -> 0 -> 6 -> 5
		y = 1
		z = 10 -> 11 -> 7
		procedure A
		function t
		procedure B
		Valor de retorno
		
		*2 ***Registro de Activación A
		Punto de Retorno
		Enlace estático (*1)
		Enlace dinámico (*1)
		t = 1
		y = 2 -> 3
		Valor de Retorno
		
		*3 ***Registro de Activación B
		Punto de Retorno
		Enlace estático (*1)
		Enlace dinámico (*2)
		d = 0 -> 2 -> 4 -> 6
		procedure I
		Valor de Retorno

```
    while x>d do begin
      I(); x:=x-1;
      d:=d + 2;
    end;
  end;
begin
  For x:=1 To 10 do a(x):=x;
  x:=5; y:=1; z:=10;
  A();
  For x:=1 To 10 do write(a(x),x);
end.
```

*4 ***Registro de Activación t
Punto de Retorno
Enlace Estático (*1)
Enlace Dinámico (*3)
Valor de Retorno

*5 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

*6 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

*7 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

Imprime:
2,1
5,2
3,3
4,4
5,5
6,6
7,7
8,8
9,9
10,10

*4 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

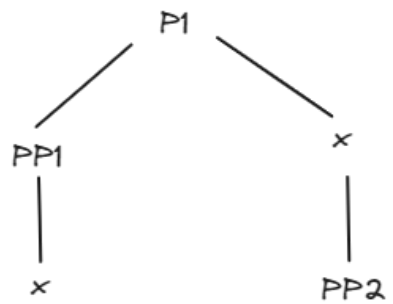
*5 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

*6 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

*7 ***Registro de Activación I
Punto de Retorno
Enlace Estático (*3)
Enlace Dinámico (*3)
Valor de Retorno

Imprime:
2,1
5,2
3,3
9,4
5,5
6,6
7,7
8,8
9,9
12,10

Ejercicio 3

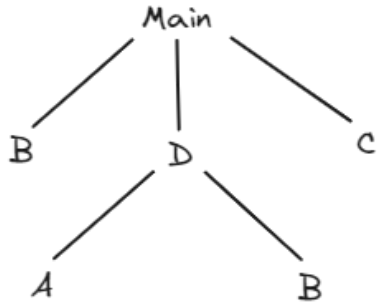


Código	Cadena estática	Cadena dinámica

PROGRAM P1;
var
a:integer;
b:char;
c: array[1..10] of integer
 	
Procedure PP1;
var
a:char;
p:integer;
Function x: integer;
var
z:integer;
begin
a:="j";
z=-1;
return z;
end;
Begin
p:=x;
write(a);
p:=x+3;
c[p]=8;
p:=x+2;
c[p]=x;
end;
 	
Procedure x;
var
b:char;
Procedure PP2;
Begin
write("para qué estoy aquí?");
end;
Begin
a:=1;
c[a]:=4;
b:="a";
	*1 ***Registro de Activación P1
	Punto de Retorno
	a = 3 -> 3...10 -> 1 -> 1...10
	b = "c"
	c(1) = 4 -> -1
	c(2) = 8
	c(3) = 6
	c(4) = 8
	c(5) = 10
	c(6) = 12
	c(7) = 14
	c(8) = 16
	c(9) = 18
	c(10) = 20
	Procedure x
	Procedure PP1
	Valor de Retorno
	
	*2 ***Registro de Activación x
	Punto de Retorno
	Enlace estático (*1)
	Enlace dinámico (*1)
	b = "a" -> "b"
	Procedure PP2
	Valor de Retorno
	
	*3 ***Registro de Activación PP1
	Punto de Retorno
	Enlace estático (*1)
	Enlace dinámico (*2)
	a = "j"
	p = -1 -> 2 -> 1
	Valor de Retorno -1 -> -1 -> -1 -> -1
	
	*4 ***Registro de Activación x
	Punto de Retorno
	Enlace estático (*3)
	Enlace dinámico (*3)
	z = -1
	Valor de Retorno
	
	*5 ***Registro de Activación x
	Punto de Retorno
	Enlace estático (*3)
	Enlace dinámico (*3)
	z = -1
	Valor de Retorno
	
	*6 ***Registro de Activación x
		*1 ***Registro de Activación P1
		Punto de Retorno
		a = 3 -> 3...10 -> 1 -> 1...10
		b = "c"
		c(1) = 4 -> -1
		c(2) = 8
		c(3) = 6
		c(4) = 8
		c(5) = 10
		c(6) = 12
		c(7) = 14
		c(8) = 16
		c(9) = 18
		c(10) = 20
		Procedure x
		Procedure PP1
		Valor de Retorno
		
		*2 ***Registro de Activación x
		Punto de Retorno
		Enlace estático (*1)
		Enlace dinámico (*1)
		b = "a" -> "b"
		Procedure PP2
		Valor de Retorno
		
		*3 ***Registro de Activación PP1
		Punto de Retorno
		Enlace estático (*1)
		Enlace dinámico (*2)
		a = "j"
		p = -1 -> 2 -> 1
		Valor de Retorno -1 -> -1 -> -1 -> -1
		
		*4 ***Registro de Activación x
		Punto de Retorno
		Enlace estático (*3)
		Enlace dinámico (*3)
		z = -1
		Valor de Retorno
		
		*5 ***Registro de Activación x
		Punto de Retorno
		Enlace estático (*3)
		Enlace dinámico (*3)
		z = -1
		Valor de Retorno
		
		*6 ***Registro de Activación x

write(concat(c[1],b)); /*concat convierte a string los parámetros, concatena y retorna un string;*/
PP1();
b:="b";
write(concat(c[5],b)); /*concat convierte a string los parámetros, concatena y retorna un string;*/
End;
 	
BEGIN
a:=3;
b:="c";
for a:=3 to 10 do
begin
c[a]:=2*a;
end;
x;
write(b);
write(a);
for a:=1 to 10 do
write(c[a]-3);
END.
	Punto de Retorno
	Enlace estático (*3)
	Enlace dinámico (*3)
	z = -1
	Valor de Retorno
	
	*7 ***Registro de Activación x
	Punto de Retorno
	Enlace estático (*3)
	Enlace dinámico (*3)
	z = -1
	Valor de Retorno
	
	Imprime:
	4a
	j
	10b
	c
	1
	-4
	5
	3
	5
	7
	9
	11
	13
	15
	17
		Punto de Retorno
		Enlace estático (*3)
		Enlace dinámico (*3)
		z = -1
		Valor de Retorno
		
		*7 ***Registro de Activación x
		Punto de Retorno
		Enlace estático (*3)
		Enlace dinámico (*3)
		z = -1
		Valor de Retorno
		
		Imprime:
		4a
		j
		10b
		c
		1
		-4
		5
		3
		5
		7
		9
		11
		13
		15
		17

Ejercicio 4



Código	Cadena estática	Cadena dinámica
Procedure Main; var x, y: integer; vec: array[1..7] of integer;	*1 Registro de Activación Main Punto de Retorno x = 1...7 -> 3 -> 2 -> 2 -> 4 -> 2 -> 2 -> 1...7 vec(1) = 1 -> 2	*1 Registro de Activación Main Punto de Retorno x = 1...7 -> 3 -> 2 -> 1...7 vec(1) = 1 vec(2) = 2 -> 4

```
Function B:integer;
   var y:integer;
begin
   y:=4; x:= y - 2;
   return (x);
end;

Procedure D;
   var i, x: integer;
   vec: array[1..7] of integer;

   Procedure A;
      var y:integer;
   begin
      y:=x + 5; vec(i + 2):= vec(i + 2) + y;
      x:= x +B; C;
   end;

   Function B:integer;
   begin
      vec(i):= y + 2; i:=i+2;
      vec(i):= vec(1) * i;
      return ( vec(i)-vec(1) );
   end;

begin
   for x:= 1 to 7 do vec(x):= 1;
   x:=1; i:= 2;
   if y = 7 then A; else C;
   for x:= 1 to 7 do write(vec(x));
end;

Procedure C;
   var i, y: integer;
begin
   i:= 1; y:= 6; x:= x + B;
   vec(2):= vec(2) * x;
   while (i < y) do begin
      vec(i):= vec(i) + B - 1;
```

vec(2) = 2 -> 8 -> 10
vec(3) = 3
vec(4) = 4 -> 5
vec(5) = 5 -> 30
vec(6) = 6
vec(7) = 7
y = 7
Function B
Procedure D
Procedure C
Valor de Retorno 2

*2 Registro de Activación B
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*1)
y = 4
valor de Retorno

*3 Registro de Activación D
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*1)
x = 1...7 -> 1 -> 4 -> 1...7
vec(1) = 1
vec(2) = 1 -> 9
vec(3) = 1
vec(4) = 1 -> 7 -> 4
vec(5) = 1
vec(6) = 1
vec(7) = 1
i = 2 -> 4
Procedure A
Function B
Valor de Retorno

*4 Registro de Activación A
Punto de Retorno
Enlace estático (*3)
Enlace dinámico (*3)
y = 6
Valor de Retorno 3

*5 Registro de Activación B
Punto de Retorno
Enlace estático (*3)
Enlace dinámico (*4)
Valor de Retorno

*6 Registro de Activación C
Punto de Retorno

vec(3) = 3
vec(4) = 4
vec(5) = 5 -> 12
vec(6) = 6
vec(7) = 7
y = 7
Function B
Procedure D
Procedure C
Valor de Retorno 2

*2 Registro de Activación B
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*1)
y = 4
Valor de Retorno

*3 Registro de Activación D
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*1)
x = 1...7 -> 1 -> 4 -> 20 -> 1...7
vec(1) = 1 -> 8
vec(2) = 1 -> 8 -> 160
vec(3) = 1 -> 24 -> 8
vec(4) = 1 -> 7 -> 4
vec(5) = 1 -> 40 -> 71
vec(6) = 1
vec(7) = 1
i = 2 -> 4
Procedure A
Function B
Valor de Retorno

*4 Registro de Activación A
Punto de Retorno
Enlace estático (*3)
Enlace dinámico (*3)
y = 6
Valor de Retorno 3

*5 Registro de Activación B
Punto de Retorno
Enlace estático (*3)
Enlace dinámico (*4)
Valor de Retorno

*6 Registro de Activación C
Punto de Retorno
Enlace estático (*1)

```
        i:= i + 3;
      end;
      y:= y - 4;
   end;

begin
   for x:= 1 to 7 do vec(x):= x;
   x:= 3; y:= B+5; D;
   if (x = 2) then begin
      vec(x):= vec(x) + 2;
      vec(x + 3):= vec(x) * 3;
   end;
   for x:= 1 to 7 do write(vec(x));
end.
```

Enlace estático (*1)
Enlace dinámico (*4)
i = 1 -> 4 -> 7
y = 6 -> 2
Valor de Retorno 2 -> 2 -> 2

*7 Registro de Activación B
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*6)
y = 4
valor de Retorno

*8 Registro de Activación B
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*6)
y = 4
valor de Retorno

*9 Registro de Activación B
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*6)
y = 4
valor de Retorno

Imprime:
1
9
1
4
1
1
1
2
10
3
5
30
6
7

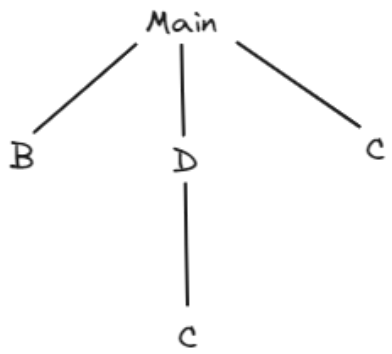
Enlace dinámico (*4)
i = 1 -> 3 -> 5 -> 8
y = 6 -> 2
Valor de Retorno 16 -> 32

*7 Registro de Activación B
Punto de Retorno
Enlace estático (*3)
Enlace dinámico (*6)
Valor de Retorno

*8 Registro de Activación B
Punto de Retorno
Enlace estático (*3)
Enlace dinámico (*6)
Valor de Retorno

Imprime:
8
160
8
4
71
1
1
1
4
3
4
12
6
7

Ejercicio 5



Código	Cadena estática	Cadena dinámica

Program Main;
Var x, y, z:integer;
a, b: array[1..6] of integer;
 	
Procedure B;
var y,x: integer;
 	
Procedure C;
var c:integer;
begin
y:= y + 2; c:=2;
a(x):=a(x)*y;
if (y >7) then
b(y-6)=b(4)*2+b(y-6);
D;
end;
 	
begin
x:=2; y:= x + 3;
C; x:= x + 1; write (x,y);
End;
 	
Procedure D;
begin
x:= c + 5 + x;
y:= y + 2;
	*1 Registro de Activación Main
	Punto de Retorno
	x = 1 -> 2 -> 11
	y = 2 -> 4
	z = 1...6 -> 1...6
	a(1) = 1
	a(2) = 2 -> 14 -> 21
	a(3) = 3
	a(4) = 4 -> 23
	a(5) = 5
	a(6) = 6
	b(1) = 3 -> 4
	b(2) = 4
	b(3) = 5
	b(4) = 6
	b(5) = 7
	b(6) = 8
	Procedure B
	Procedure D
	Function C
	Valor de Retorno
	
	*2 Registro de Activación B
	Punto de Retorno
	Enlace estático (*1)
	Enlace dinámico (*1)
	x = 2 -> 3
	y = 5 -> 7
	Procedure C
	Valor de Retorno
	
	*3 Registro de Activación C
	Punto de Retorno
	Enlace estático (*2)
		*1 Registro de Activación Main
		Punto de Retorno
		x = 1
		y = 2
		z = 1...6 -> 1...6
		a(1) = 1
		a(2) = 2 -> 14
		a(3) = 3
		a(4) = 4
		a(5) = 5
		a(6) = 6
		b(1) = 3
		b(2) = 4
		b(3) = 5
		b(4) = 6
		b(5) = 7
		b(6) = 8
		Procedure B
		Procedure D
		Function C
		Valor de Retorno
		
		*2 Registro de Activación B
		Punto de Retorno
		Enlace estático (*1)
		Enlace dinámico (*1)
		x = 2 -> 9 -> 10
		y = 5 -> 7 -> 9
		Procedure C
		Valor de Retorno
		
		*3 Registro de Activación C
		Punto de Retorno
		Enlace estático (*2)

```
  end;

  Function C: integer;
  begin
    b(x):= b(x) + 1;
    x:= x + 1;
    a(y):=a(y)+b(x)+3;
    a(x+2)=a(x) + 2;
    return b(x);
  end

begin
  x:= 1; Y:= 2;
  for z:=1 to 6 do begin
    a(z):= z;
    b(z):= z + 2;
  end;
  B;
  for z:=1 to 6 do write (a(z), b(z));
end.
```

Enlace dinámico (*2)
c = 2
Valor de Retorno

*4 Registro de Activación D
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*3)
Valor de Retorno 4

*5 Registro de Activación C
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*4)
Valor de Retorno

Imprime:
3,7
1,4
21,4
3,5
23,6
5,7
6,8

Enlace dinámico (*2)
c = 2
Valor de Retorno

*4 Registro de Activación D
Punto de Retorno
Enlace estático (*1)
Enlace dinámico (*3)
Valor de Retorno

Imprime:
10,9
1,3
14,4
3,5
4,6
5,7
6,8

Inciso c)

- No es lo mismo en la estática ya que ahí es una función que modifica x, pero en la dinámica c se reconoce como una variable.