Práctica 5 CPLP

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

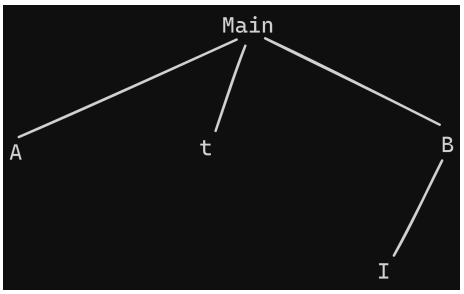
Ejercicio 1:

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

Registro de activación	Explicación
Head (prog principal)	Current: Dirección base del registro de activación actual. Free: Próxima dirección libre en la pila.
Pto retorno	Instrucción a la que se vuelve después de ejecutar una subrutina.
EE (enlace estático)	Apunta al registro de activación de la rutina que lo contiene estaticamente.
ED (enlace dinámico)	Apunta a la dirección base del registro de activación de la rutina que lo llama.
Variables...	Variables definidas dentro de la rutina
...	
Parámetros...	Variables a ser usadas por la siguiente rutina.
...	
Procedimientos...	Procedimientos definidos dentro de la unidad.
...	
Funciones...	Funciones definidas dentro de la unidad.
...	
Valor de retorno	Valor de retorno de una subrutina llamada dentro de la unidad, ya que cuando termina la rutina se desaloja, se pone aca.

La utilidad del registro es guardar de manera estructurada en memoria toda la información necesaria para la ejecución de una unidad de código.

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




```
1.    Program Main
2.        Var a: array[1..10] of integer;
3.            x,y,z:integer
4.        Procedure A ()
5.            var y,t: integer;
6.            begin
7.            a(1):= a(1)+1;z:=z+1;
8.            t:=1; y:=2;
9.            B(); a(y):=a(y)+3; y:=y+1;
10.           If z=11 Then Begin
11.           a(z-1):=a(z-2) + 3;
12.           z:=z-4;
13.           a(z-y):=a(z) - a(y) + 5;
14.       End;
15.   end;
16.   Function t():integer
17.       begin
18.           y:=y+1; z:=z-6;
19.           return(y+x);
20.       end;
21.   Procedure B()
22.       var d:integer;
23.       Procedure I ()
24.       begin
25.           x:=0; x:=x+6;
26.       end;
27.       begin
28.           x:=x+t; d:=0;
29.           while x>d do begin
30.               I(); x:=x-1;
31.               d:=d + 2;
```

Cadena estática	
	***** Reg Activ Main**
*1	Pto 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 -> 13 -> 0 -> 6 -> 5 -> 0 -> 6 ->5 -> 0 -> 6 ->5 -> 1..10
	Y= 1 –> 2
	Z=10 –> 11 –> 5
	Procedure A
	Function T
	Procedure B

```
32.           end;
33.       end;
34.   begin
35.       For x:=1 To 10 do a(x):=x;
36.       x:=5; y:=1; z:=10;
37.       A();
38.       For x:=1 To 10 do write(a(x),x);
39.   end.
```

	VR . . . . . .
***2**	******Reg Activ A**
	Pto Retorno
	EE (*1)
	ED (*1)
	Y = 2 -> 3
	T = 1
	VR . . . .
*3	*** Reg Activ B
	Pto Retorno
	EE(*1)
	ED(*2)
	D = 0 -> 2 -> 4 -> 6
	Procedure I
	VR . . 8 . .
*4	***** Reg Activ t**
	Pto retorno
	EE(*1)
	ED(*3)
	VR
	----
*5	***** Reg Activ I**
	Pto retorno
	EE(*3)
	ED(*3)
	VR
	----
*5	***** Reg Activ I**

	Pto retorno
	EE(*3)
	ED(*3)
	VR
	----
*5	***** Reg Activ I**
	Pto retorno
	EE(*3)
	ED(*3)
	VR

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

Cadena dinamica	
	***** Reg Activ Main**
*1	Pto 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 -> 1...10
	Y= 1 –> 2
	Z=10 –> 11 -> 7
	Procedure A
	Function T
	Procedure B
	VR . . . . . .
***2**	******Reg Activ A**
	Pto Retorno
	EE (*1)
	ED (*1)
	Y = 2 -> 3
	T = 1
	VR . . . .
*3	*** Reg Activ B
	Pto Retorno
	EE(*1)
	ED(*2)
	D = 0 -> 2 -> 4 -> 6
	Procedure I
	VR . . . .
*4	***** Reg activ I**
	Pto retorno
	EE(*3)

	ED(*3)
	VR
	--

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

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

```
1.    PROGRAM P1;
2.    var
3.        a:integer;
4.        b:char;
5.        c: array[1..10] of integer
6.    Procedure PP1;
7.    var
8.        a:char;
9.        p:integer;
10.       Function x: integer;
11.       var
12.           z:integer;
13.       begin
14.           a:="j";
15.           z=-1;
16.           return z;
17.       end;
18.   Begin
19.       p:=x;
```

Cadena estática	
*1	***** Reg activ P1**
	P.R
	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

```
20.         write(a);
21.         p:=x+3;
22.         c[p]=8;
23.         p:=x+2;
24.         c[p]=x;
25.     end;
26.     Procedure x;
27.     var
28.         b:char;
29.         Procedure PP2;
30.         Begin
31.             write("para qué estoy
        aquí?");
32.         end;
33.     Begin
34.         a:=1;
35.         c[a]:=4;
36.         b:="a";
37.         write(concat(c[1],b)); /*concat
        convierte a string los
38.         parámetros, concatena y retorna
        un string;*/
39.         PP1();
40.         b:="b";
41.         write(concat(c[5],b)); /*concat
        convierte a string los
42.         parámetros, concatena y retorna
        un string;*/
43.     End;
44.     BEGIN
45.         a:=3;
46.         b:="c";
47.         for a:=3 to 10 do
48.             begin
49.                 c[a]:=2*a;
50.             end;
51.         x;
52.         write(b);
53.         write(a);
54.         for a:=1 to 10 do
55.             write(c[a]-3);
56.     END.
```

	c(8) = 16
	c(9) = 18
	c(10) = 20
	Procedure PP1
	Procedure X
	VR
*2	***** Reg activ x**
	P.R
	EE(*1)
	ED(*1)
	b = “a” -> “b”
	Procedure PP2
	VR
*3	***** Reg activ PP1**
	P.R
	EE(*1)
	ED(*2)
	a = “j” -> “j” -> “j” -> “j”
	p = -1 -> 2 -> 1
	Function x
	V.R -1 -> -1 -> -1 -> -1
*4	***** Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R
*5	***** Reg activ X**

	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R
*6	***** Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R
*7	***** Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R

Imprime:
4a
j
10b
c
1
-4
5
3
5
7
9
11
13
15
17

Cadena dinámica (es igual q la estática)

*1	***** Reg activ P1**
	P.R
	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
	V.R
*2	**Reg activ X**
	P.R
	EE(*1)
	ED(*1)
	b = “a” -> “b”
	Procedure PP2
	V.R
*3	**Reg activ PP1**
	P.R
	EE(*1)
	ED(*2)

	a = “j” -> “j” -> “j” -> “j”
	p = -1 -> 2 -> 1
	Function x
	V.R -1 -> -1 -> -1 -> -1
*4	**Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R
*5	**Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R
*6	**Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R
*7	**Reg activ X**
	P.R
	EE(*3)
	ED(*3)
	z = -1
	V.R

	Imprime 4a j 10a c 1 -4 5 3 5 7 9 11 13

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
// Elijo creer. El dinámico lo hice como 4 veces porque estaba quemado.

Código

```
1.  Procedure Main;
2.     var x, y: integer;
3.     vec: array[1..7] of integer;
4.     Function B:integer;
5.        var y:integer;
6.     begin
7.        y:=4; x:= y - 2;
8.        return (x);
9.     end;
10.    Procedure D;
11.       var i, x: integer;
12.       vec: array[1..7] of integer;
13.       Procedure A;
14.          var y:integer;
15.       begin
16.          y:=x + 5; vec(i + 2):=
17.          vec(i + 2) + y;
18.          x:= x +B; C;
19.       end;
20.       Function B:integer;
21.       begin
22.          vec(i):= y + 2; i:=i+2;
23.          vec(i):= vec(1) * i;
24.          return ( vec(i)-vec(1) );
25.       end;
```

Cadena estática

*1	Reg main
	P.R
	X = 1..7 -> 3 -> 2 -> 2 -> 4 -> 2 -> 2 -> 1..7
	Y = 7
	Vec(1) = 1 -> 2
	Vec(2) = 2 -> 8 -> 10
	Vec(3) = 3
	Vec(4) = 4 -> 5
	Vec(5) = 5 -> 30
	Vec(6) = 6
	Vec(7) = 7
	Function B

Cadena dinámica

*1 REG Main
P.R
X = 1..7 -> 3 -> 2 -> 1..7
Y = 7
VEC(1)= 1
VEC(2)= 2 -> 4
VEC(3)= 3
VEC(4)= 4
VEC(5)= 5 -> 12
VEC(6)= 6
VEC(7)= 7
FUNCTION B
PROCEDURE D
PROCEDURE C
V.R = 2

*2 REG B (SUPERIOR)
P.R
EE (*1)
ED (*1)
Y = 4
V.R
*3 REG D
P.R
EE(*1)

```
26.     begin
27.       for x:= 1 to 7 do vec(x):= 1;
28.       x:=1; i:= 2;
29.       if y = 7 then A; else C;
30.       for x:= 1 to 7 do write(vec(x));
31.     end;
32.     Procedure C;
33.       var i, y: integer;
34.     begin
35.       i:= 1; y:= 6; x:= x + B;
36.       vec(2):= vec(2) * x;
37.       while (i < y) do begin
38.         vec(i):= vec(i) + B - 1;
39.         i:= i + 3;
40.       end;
41.       y:= y - 4;
42.     end;
43.     begin
44.       for x:= 1 to 7 do vec(x):= x;
45.       x:= 3; y:= B+5; D;
46.       if (x = 2) then begin
47.         vec(x):= vec(x) + 2;
48.         vec(x + 3):= vec(x) * 3;
49.       end;
50.       for x:= 1 to 7 do write(vec(x));
51.   end.
```

	Procedure D
	Procedure C
	V.R 2
*2	Reg activ B
	P.R
	EE(*1)
	ED(*1)
	Y= 4
	V.R
*3	Reg activ D
	P.R
	EE(*1)
	ED(*1)
	I = 2 -> 4
	X= 1..7 -> 1 -> 4 -> 1..7
	Vec(1) = 1 -> 4
	Vec(2) = 1 -> 9
	Vec(3) = 1
	Vec(4) = 1 -> 7
	Vec(5) = 1
	Vec(6) = 1
	Vec(7) = 1
	Procedure A
	Function B
	V.R
*4	Reg activ A
	P.R
	EE(*3)

```
   ED(*1)
   I = 2 -> 4
   X = 1..7 -> 1 -> 4 -> 20
   VEC(1)= 1 -> 8
   VEC(2)= 1 -> 8 -> 160
   VEC(3)= 1 -> 24 -> 8 -> 39
   VEC(4)= 1 -> 7 -> 4
   VEC(5)= 1 -> 40
   VEC(6)= 1
   VEC(7)= 1
   PROCEDURE A
   FUNCTION B
   V.R
*4 REG A
   P.R
   EE(*3)
   ED(*3)
   Y= 6
   V.R 3
*5 REG B (INFERIOR)
   P.R
   EE(*3)
   ED(*4)
   V.R
*6 REG C
   P.R
   EE(*1)
   ED(*4)
   I = 1 -> 3 -> 5 -> 8
   Y = 6 -> 2
   V.R 16 -> 32
*7 REG B(INFERIOR)
   P.R
   EE(*3)
   ED(*6)
   V.R
*8 REG B(INFERIOR)
   P.R
   EE(*3)
   ED(*6)
   V.R
```

Imprimimos:
8
160
39
4
40
1
1

	ED(*3)
	Y = 6
	V.R 3
*5	Reg activ B
	P.R
	EE(*3)
	ED(*4)
	V.R
*6	Reg activ C
	P.R
	EE(*1)
	ED(*4)
	I= 1 -> 4 -> 7
	Y = 6 -> 2
	V.R 2 -> 2 -> 2
*7	Reg activ B
	P.R
	EE(*1)
	ED(*6)
	Y = 4
	V.R
*8	Reg activ B
	P.R
	EE(*1)
	ED(*6)
	Y = 4
	V.R
*9	Reg activ B

1
4
3
4
12
6
7

	P.R
	EE(*1)
	ED(*6)
	Y = 4
	V.R

Imprime:
4
9
1
7
1
1
1

2
10
3
5
30
6
7

Código	Estático	Dinámico

```
1.    Program Main;
2.        Var x, y, z:integer;
3.        a, b: array[1..6] of
      integer;
4.        Procedure B;
5.            var y,x: integer;
6.            Procedure C;
7.            var c:integer;
8.            begin
9.                y:= y + 2;
      c:=2;
10.               a(x):=a(x)*y;
11.               if (y >7) then
12.
      b(y-6)=b(4)*2+b(y-6);
13.               D;
14.           end;
15.           begin
```

*1 REG ACTIV MAIN
P.R
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
V.R

```
16.                 x:=2; y:= x +
    3;
17.                 C; x:= x + 1;
    write (x,y);
18.             End;
19.         Procedure D;
20.         begin
21.             x:= c + 5 + x;
22.             y:= y + 2;
23.         end;
24.         Function C: integer;
25.         begin
26.             b(x):= b(x) + 1;
27.             x:= x + 1;
28.             a(y):=a(y)+b(x)+3;
29.             a(x+2)=a(x) + 2;
30.             return b(x);
31.         end
32.     begin
33.         x:= 1; Y:= 2;
34.         for z:=1 to 6 do begin
35.             a(z):= z;
36.             b(z):= z + 2;
37.         end;
38.         B;
39.         for z:= to 6 do write
    (a(z), b(z));
40.     end.
```

*2 REG ACTIV B
P.R
EE(*1)
ED(*1)
Y= 5 -> 7
X= 2 -> 3
PROCEDURE C
V.R
*3 REG ACTIV C (INFERIOR)
P.R
EE(*2)
ED(*2)
C=2
V.R
*4 REG ACTIV D
P.R
EE(*1)
ED(*3)
V.R 4
*5 REG ACTIV C (SUPERIOR)
P.R
EE(*1)
ED(*4)
V.R

IMPRIME:
3,7
1
21
3
23
5
6

4
4
5
6
7
8