Sesión 1 – Estructura de un programa

Pru1.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure pru1 is
begin
   Put ("hola");
end pru1;
```

pru2.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;
with elotro;
with paquete;

procedure pru2 is
begin
   Put ("hola");
   elotro;
   paquete.p1;
   paquete.p2;
end pru2;
```

elotro.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure elotro is
begin
   Put ("-- soy el otro");
end elotro;
```

paquete.ads

```
package paquete is
  procedure p1 ;
  procedure p2 ;
end paquete;
```

paquete.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

package body paquete is

  procedure p1 is
  begin
     Put ("... se ejecuta paquete.p1 ");
  end p1;

  procedure p2 is
  begin
     Put ("... se ejecuta paquete.p2 ");
  end p2;

begin
   Put ("-- parte principal del paquete");
end paquete;
```

Sesión 2 – Estructuras de control

```
Bucles.adb
```

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure bucles is

  type Tipo_Vector is array (1..20) of integer;

  Mi_Tabla: Tipo_Vector := (2,2,4,1,6,6,7,9,5,6,8,3,3,0,4,8,9,6,4,6);
  i : integer range 1..20;
  suma: integer := 0;

begin
  Put ("Inicio del recorrido del Vector: ");
  New_line;

  for j in 1..20 loop
    Put(integer'Image(Mi_Tabla(j))); Put(" ");
  end loop;
  New_Line;

  i:=1;
  loop
    Put(integer'Image(Mi_Tabla(i))); Put(" ");
    exit when Mi_Tabla(i)=0;
    i:=i+1;
  end loop;
  New_line;

  i :=1;
  Put(integer'Image(suma));
  while (suma < 20) loop
    suma := suma + Mi_Tabla(i);
    Put (" +");
    Put(integer'Image(Mi_Tabla(i)));
    Put(" =");
    Put(integer'Image(suma));
    i := i + 1;
  end loop;
  New_Line;

 for j in 1..20 loop
    Put(integer'Image(Mi_Tabla(j)));
    if ((Mi_Tabla(j) rem 2) = 0) then
      Put ("=Par ");
    else
      Put ("=Impar ");
    end if;
  end loop;
  New_Line;

end;
```

Sesión 3 – Tipos de datos

```
Tipos1.adb
```

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure tipos1 is

-- Tipos y subtipos enumerados

type dias_semana is (lunes,martes,miercoles,jueves,viernes,sabado,domingo);

type entresemana is (lunes,martes,miercoles,jueves,viernes);

subtype dias_laborables is dias_semana range lunes..viernes;
subtype dias_findesemana is dias_semana range sabado..domingo;

hoy, otro_dia: dias_semana;
reunion_1: entresemana;
reunion_2: dias_laborables;
teatro: dias_findesemana;

begin

  hoy := lunes;
  otro_dia := domingo;

  reunion_1 := martes;
  reunion_2 := miercoles;
  teatro := sabado;

  -- reunion_1:= hoy; -- error compilacion!!
  reunion_2:= hoy; -- correcto
  teatro := otro_dia;

  Put ("Se imprimen los valores: ");
  New_line;

  Put (entresemana'Image(reunion_1)); New_Line;
  Put (dias_laborables'Image(reunion_2)); New_Line;
  Put (dias_findesemana'Image(teatro)); New_Line;


end;
```

agua_recogida.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure agua_recogida is

subtype Dia_del_mes is Integer range 1..31;

type Agua_recogida is array (Dia_del_mes) of integer;

type numero_real is digits 3 range 0.0..100.0; -- coma flotante

Enero : Agua_recogida := (15=>20, 16=>40, 17=>30, others=>0);

i : Dia_del_mes;
suma: integer := 0;
media: float := 0.0;
media_aritmetica: numero_real := 0.0;

begin
  New_Line;
  Put ("Agua recogida");
  New_line;
  Put ("Enero:");
  i:=1;
  loop
    Put(integer'Image(Enero(i)));
    suma := suma + Enero(i);
    exit when i=31;
    i:=i+1;
  end loop;
  New_line;

  Put ("Suma total = ");
  Put (integer'Image(suma)); New_line;

  Put ("Media operando con enteros = ");
  media := float(suma / dia_del_mes'Last);
  Put (float'Image(media)); New_line;

  media := 0.0;
  Put ("Media con operandos en coma flotante = ");
  media := (float(suma) / float(dia_del_mes'Last));
  Put (float'Image(media)); New_line;

  Put ("Media limitanto el numero de decimales = ");
  media_aritmetica := (numero_real(suma) / numero_real(dia_del_mes'Last));
  Put (numero_real'Image(media_aritmetica)); New_line;

  media_aritmetica := 0.0;
  Put ("Media dividiendo directamente entre 31 = ");
  -- media_aritmetica := numero_real(Suma) / 31; -- Error de compilacion!!
  Put (numero_real'Image(media_aritmetica)); New_line;
end;
```

```
tiras.adb
```

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure tiras is

-- Tiras de caracteres o strings

S: String (1..20) := (others => '-');

Estrellas: String (1..200) := (1..200 => '*');
Str1: String (1..8) := "un valor";
Str2: String (3..10);
Str3: String (1..10);

begin
  New_line;
  Put ("Imprimimos las tiras de caracteres"); New_Line;
  Put (Str1); New_Line;

  Str2 := Str1;
  Put (Str2); New_Line;

  Str3 := Estrellas(21..30);
  Put (Str3); New_Line;

  S := "literal + " & Str3;
  Put (S); New_Line;
end;
```

Sesión 4 – Procedimientos

```
procedimientos.adb
```

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure procedimientos is

subtype Dia_del_mes is Integer range 1..31;
type Agua_recogida is array (Dia_del_mes) of integer;

-----------------------------------------------------
-- Declaraciones
-----------------------------------------------------
procedure Incrementar (d: in integer);
procedure Modificar (elemento: in Dia_del_Mes; d: in integer; ant: out integer);
function Leer (j: in Dia_del_Mes) return integer;

Enero : Agua_recogida := (15=>20, 16=>40, 17=>30, others=>0);
valor_anterior: integer;

--------------------------------------------------------
-- Procedimientos
--------------------------------------------------------
  procedure Incrementar (d: in integer) is
  begin
    for i in Dia_del_Mes loop
      Enero (i) := Enero (i) + d;
    end loop;
  end Incrementar;

  procedure Modificar (elemento: in Dia_del_Mes; d: in integer; ant: out integer) is
  begin
    ant := Enero (elemento);
    Enero (elemento) :=  d;
    Put ("Modificado elemento ");
    put(Dia_del_Mes'Image(elemento));
    Put (" con nuevo valor ");
    put(integer'Image(d));
  end Modificar;

  function Leer (j: in Dia_del_Mes) return integer is
  begin
    return Enero (j);
  end Leer;
```

```
--------------------------------------------------------
-- Cuerpo del programa principal
--------------------------------------------------------
begin
  New_line;
  Put ("Agua recogida durante mes de Enero");
  New_line;

  Incrementar(2);

  Modificar(2,3,valor_anterior);
  Put (" valor anterior ");
  Put(integer'Image(valor_anterior));
  New_Line;

  Modificar(4,11,valor_anterior);
  Put (" valor anterior ");
  Put(integer'Image(valor_anterior));
  New_Line;

  declare -- Esto es un bloque
   k:integer;
  begin
    Put_Line ("Valores finales ");
    for i in Dia_del_Mes loop
      k:=Leer(i);
      put(Integer'Image(k));
    end loop;
  end;  -- Fin del bloque


end procedimientos;
```

Pru5.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;
with paquete; -- use paquete;

procedure pru5 is

valor_anterior: integer;

begin
  Put ("Comienza pru5 ");
  New_Line;

  Put ("Agua recogida durante mes de Enero");
  New_line;

  paquete.Incrementar(2);

  paquete.Modificar(2,3,valor_anterior);
  Put (" valor anterior ");
  Put(integer'Image(valor_anterior));
  New_Line;

  paquete.Modificar(4,11,valor_anterior);
  Put (" valor anterior ");
  Put(integer'Image(valor_anterior));
  New_Line;

  declare -- Esto es un bloque
   k:integer;
  begin
    Put_Line ("Valores finales ");
    for i in paquete.Dia_del_Mes loop
      k:= paquete.Leer(i);
      put(Integer'Image(k));
    end loop;
  end;  -- Fin del bloque

end pru5;
```

Paquete.ads

```
package paquete is

-----------------------------------------------------
-- Datos exportados
-----------------------------------------------------

subtype Dia_del_mes is Integer range 1..31;
type Agua_recogida is array (Dia_del_mes) of integer;

Enero : Agua_recogida;

-----------------------------------------------------
-- Procedimientos exportados
-----------------------------------------------------
procedure Incrementar (d: in integer);
procedure Modificar (elemento: in Dia_del_Mes; d: in integer; ant: out integer);
function Leer (j: in Dia_del_Mes) return integer;

end paquete;
```

Paquete.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

package body paquete is

--------------------------------------------------------
-- Cuerpo de los Procedimientos
--------------------------------------------------------
  procedure Incrementar (d: in integer) is
  begin
    for i in Dia_del_Mes loop
      Enero (i) := Enero (i) + d;
    end loop;
  end Incrementar;

  procedure Modificar (elemento: in Dia_del_Mes; d: in integer; ant: out integer) is
  begin
    ant := Enero (elemento);
    Enero (elemento) :=  d;
    Put ("Modificado elemento ");
    put(Dia_del_Mes'Image(elemento));
    Put (" con nuevo valor ");
    put(integer'Image(d));
  end Modificar;

  function Leer (j: in Dia_del_Mes) return integer is
  begin
    return Enero (j);
  end Leer;


begin
   Put ("-- inicializacion del paquete");
   Enero := (15=>20, 16=>40, 17=>30, others=>0);
end paquete;
```

Sesión 7 – Tareas

lanzatareas.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure lanzatareas is

      --   pragma Priority (System.Priority'First);
      task A;
      task B;

      task body A is
      begin
        Put_Line ("AAAA");
        delay (0.0);
        Put_Line ("AAAA");
      end A;

      task body B is
      begin
        Put_Line ("BBBB");
        delay (0.0);
        Put_line ("BBBB");

      end B;

begin
   Put_Line ("hola");
   delay (0.0);
   Put_Line ("adios");

end lanzatareas;
```

usapaqtareas.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;
with paqtareas;

procedure usapaqtareas is

begin
   Put_Line ("hola");
   paqtareas.p1;
   paqtareas.p2;
   Put_Line ("adios");
end usapaqtareas;
```

paqtareas.ads

```
package paqtareas is

procedure p1 ;
procedure p2 ;

task A;
task B;

end paqtareas;
```

paqtareas.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

package body paqtareas is

      procedure p1 is
      begin
         Put_Line ("... se ejecuta paquete.p1 ");
      end p1;

      procedure p2 is
      begin
         Put_Line  ("... se ejecuta paquete.p2 ");
      end p2;


      task body A is
      begin
        Put_Line ("AAAA");
        delay (0.0);
        Put_Line ("AAAA");
      end A;

      task body B is
      begin
        Put_Line ("BBBB");
        delay (0.0);
        Put_line ("BBBB");

      end B;

begin
   Put_Line  ("-- parte principal del paquete de tareas");

end paqtareas;
```

pruop.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure pruop is

subtype Dia_del_mes is Integer range 1..31;
type Agua_recogida is array (Dia_del_mes) of integer;

-----------------------------------------------------
-- Declaraciones
-----------------------------------------------------
procedure Incrementar (d: in integer);
procedure Modificar (d: integer);
function Leer (j: in Dia_del_Mes) return integer;

Enero : Agua_recogida := (15=>20, 16=>40, 17=>30, others=>0);

task A;
task B;
task C;

-----------------------------------------------------
-- Tareas
-----------------------------------------------------
task body A is
 begin
   Incrementar(2);
 end A;

task body B is
 begin
   Modificar(3);
 end B;

task body C is
 k:integer;
 begin
   delay (3.0);
   for i in Dia_del_Mes loop
    k:=Leer(i);
    put(Integer'Image(k));
   end loop;
 end C;
```

```
--------------------------------------------------------
-- Procedimientos
--------------------------------------------------------
  procedure Incrementar (d: in integer) is
  begin
    for i in 1..15 loop
      Enero (i) := Enero (i) + d;
    end loop;
    delay (0.0);
    for i in 15..Dia_del_Mes'last loop
      Enero (i) := Enero (i) + d;
    end loop;
  end Incrementar;

  procedure Modificar (d: in integer) is
  begin
    for i in Dia_del_Mes loop
      Enero (i) :=  d;
    end loop;
  end Modificar;

  function Leer (j: in Dia_del_Mes) return integer is
  begin
    return Enero (j);
  end Leer;

--------------------------------------------------------
-- Cuerpo del programa principal
--------------------------------------------------------
begin
  New_line;
  Put ("Agua recogida durante mes de Enero");
  New_line;
end pruop;
```

lluvia.adb

```
with Kernel.Serial_Output; use Kernel.Serial_Output;

procedure lluvia is

subtype Dia_del_mes is Integer range 1..31;
type Agua_recogida is array (Dia_del_mes) of integer;

Protected Datos is
  procedure Incrementar (d: in integer);
  procedure Modificar (d: integer);
  function Leer (j: in Dia_del_Mes) return integer;
private
  Enero : Agua_recogida := (15=>20, 16=>40, 17=>30, others=>0);
end Datos;

task A;
task B;
task C;

task body A is
 begin
   Datos.Incrementar(2);
 end A;

task body B is
 begin
   delay 0.0;
   Datos.Modificar(3);
 end B;

task body C is
 k:integer;
 begin
   delay (0.5);
   for i in Dia_del_Mes loop
    k:=Datos.Leer(i);
    put(Integer'Image(k));
   end loop;
 end C;

protected body Datos is
  procedure Incrementar (d: in integer) is
  begin
    for i in Dia_del_Mes loop
      Enero (i) := Enero (i) + d;
    end loop;
  end Incrementar;

  procedure Modificar (d: in integer) is
  begin
    for i in Dia_del_Mes loop
      Enero (i) :=  d;
    end loop;
  end Modificar;

  function Leer (j: in Dia_del_Mes) return integer is
  begin
    return Enero (j);
  end Leer;

end Datos;
```

```
begin
  Put ("Agua recogida durante mes de Enero");
  New_line;
end lluvia;
```