




14 DE MARZO DE 2019

RELACIÓN 1. OCULTACIÓN DE INFORMACIÓN EN ADA

SISTEMAS DE TIEMPO REAL

LAURA AGUILERA CHECA
IGNACIO AGUILERA GÓMEZ
GRADO EN INGENIERÍA INFORMÁTICA
Universidad de Almería



Contenido

1. Autores	2
2. Introducción.....	2
3. Actividades a realizar	2
3.1. Ejercicio 1 – TIPOS DE DATOS ENTEROS	2
3.2. Ejercicio 2 – TIPOS DE DATOS DISCRETOS	3
3.3. Ejercicio 3 – TIPOS DE DATOS REALES	3
3.4. Ejercicio 4 - ARRAYS.....	3
3.5. Ejercicio 5 – CADENAS	6
3.6. Ejercicio 6 – REGISTROS.....	6
3.7. Ejercicio 7 – TIPOS DINÁMICOS.....	7

1. Autores

Apellidos	Aguilera	Checa
Nombre	Laura	
Titulación	Grado de Ingeniería Informática	

Apellidos	Aguilera	Gómez
Nombre	Ignacio	
Titulación	Grado de Ingeniería Informática	

2. Introducción

Con estos ejercicios se pretende realizar un acercamiento del alumno a la programación en ADA. Para ello, se plantean varios ejercicios sencillos de utilización de tipos de datos. Dichos ejercicios deberán de ser remitidos al profesor a través del módulo de Tareas de la plataforma Aula Virtual antes de la finalización del plazo de envío indicado. El formato de entrega de los ejercicios deberá de ser un documento PDF en el que se incluyan los comentarios y el código de cada uno de los problemas realizados.

3. Actividades a realizar

3.1. Ejercicio 1 – TIPOS DE DATOS ENTEROS

Escribir un programa en ADA con las siguientes características:

- Defina los tres tipos de datos siguientes:
 - A: entero dentro del intervalo [-55, 60].
`type a is new Integer range -55..60;`
 - B: real dentro del intervalo [10.00, 25.00]
`type b is new Integer range 10..25;`
 - C: entero sin signo de 1 byte.
`type c is new Integer range 0..255;`
- Inicialice tres variables (una por cada tipo de dato definido) y calcule las operaciones:

```
varA:a:=20;  
varB:b:=10;  
varC:c:=1;
```

- A + B**
`outA:Integer:=Integer(Integer(varA)+Integer(varB));`
 - A + C**
`outB:Integer:=Integer(Integer(varA)+Integer(varC));`
 - B + C**
`outC:Integer:=Integer(Integer(varB)+Integer(varC));`
- Imprima por pantalla los resultados de cada una de las operaciones anteriores.

```
begin  
  
    Put_Line(outA'Img);  
    Put_Line(outB'Img);  
    Put_Line(outC'Img);  
  
end Ejercicio1;
```

Resultado: 30 21 11

3.2. Ejercicio 2 – TIPOS DE DATOS DISCRETOS

Escribir un programa en ADA que defina un tipo de datos discreto llamado “Semáforo” que pueda tomar los valores {Rojo, Amarillo, Verde}.

Para ello, creamos un nuevo tipo Semáforo que puede tomar los valores “Rojo”, “Amarillo” y “Verde”.

```
procedure Ejercicio2 is
    type Semaforo is (Rojo, Amarillo, Verde);
begin
end Ejercicio2;
```

3.3. Ejercicio 3 – TIPOS DE DATOS REALES

Escribir un programa en ADA con las siguientes características:

- Defina los tipos de datos siguientes:
 - A: coma fija dentro del intervalo [-10,150] con precisión 0.001.
`type comaFija is delta 0.001 range -10.000 .. 150.000;`
 - B: coma flotante dentro del intervalo [-100, 100] con cuatro dígitos decimales.
`type comaFlotante is digits 4 range -100.0 .. 150.0;`
- Inicialice dos variables (una por cada tipo de dato definido) y calcule la operación A + B.
`varA : comaFija:=23.0;`
`varB : comaFlotante := 23.0;`

`output : float;`

- Imprima por pantalla el resultado de la operación anterior
`begin`

```
    output:= float(varA) + float(varB);
```

```
    Put_Line (float'Image(output));
```

```
end ejercicio3;
```

Resultado: 4.60000E+01

3.4. Ejercicio 4 - ARRAYS

Escribir un programa en ADA con las siguientes características:

- Defina los tipos de datos siguientes:
 - A: vector de 15 posiciones de datos enteros.
`type vectorA is array (Integer range 1 .. 15) of integer;`
 - B: matriz de tres dimensiones con tamaño 3x10 de datos reales.
`type matrix is array (1 .. 3, 1 .. 10, 1 .. 10) of float;`
 - C: vector de datos reales con cualquier tamaño.
`type vectorB is array(Integer range <>) of float;`
- Inicialice tres variables (una por cada tipo de dato definido) a 0.
`vecA : vectorA;`
`mat : matrix;`
`vecB : vectorB(1..10);`
- Imprimir por pantalla el resultado.

```
begin

    Put_Line("Vector de enteros de rango 1 a 15");

    For_Loop1:
    for x in Integer range 1..15 loop
        vecA (x) := 0;
        Put (Integer'Image (vecA (x)) & ", ");
    end loop For_Loop1;

    Put_Line("");
    Put_Line ("Matriz de reales 3x10x10");

    For_Loop2:
    for i in Integer range 1..3 loop
        For_Loop3:
        for j in Integer range 1..10 loop
            For_Loop4:
            for k in Integer range 1..10 loop
                mat (i, j, k) := 0.0;
                Put(float'Image (mat (i, j, k)) & ", ");
            end loop For_Loop4;
            Put_Line("");
        end loop For_Loop3;
    end loop For_Loop2;

    Put_Line("");
    Put_Line("Vector de reales sin rango");

    For_Loop5:
    for y in Integer range 1..10 loop
        vecB (y) := 0.0;
        Put(float'Image(vecB(y)) & ", ");
    end loop For_Loop5;

end ejercicio4;
```

Resultado:

Vector de enteros de rango 1 a 15

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

Matriz de reales 3x10x10

0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,
0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00, 0.00000E+00,

[illegible]

Vector de reales sin rango

0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,
0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00, 0.000000E+00,

3.5. Ejercicio 5 – CADENAS

Escribir un programa en ADA con las siguientes características:

1. Inicialice una constante de tipo cadena que contenga la expresión “TIEMPO REAL”.

```
type cadena is array (Positive range <>) of character;  
  
cad:constant cadena:="TIEMPO REAL";
```

2. Imprimir por pantalla la constante anterior.

```
begin  
  
    For_Loop1:  
    for x in Integer range 1..11 loop  
        Put (character'Image (cad(x)));  
    end loop For_Loop1;  
  
end ejercicio5;
```

Resultado: 'T' 'I' 'E' 'M' 'P' 'O' ' ' 'R' 'E' 'A' 'L' '

3.6. Ejercicio 6 – REGISTROS

Escribir un programa en ADA con las siguientes características:

1. Definir el tipo de datos “Registro” con los siguientes campos:

- Hora: cadena de texto
- Valor: entero
- Unidad: cadena de texto.

```
type Registro is  
    record  
        hora : Unbounded_String := To_Unbounded_String("00:00 am");  
        valor : integer := 1;  
        unidad: Unbounded_String := To_Unbounded_String("");  
    end record;
```

2. Inicializar una variable del tipo definido con: Hora: 13:00 pm, Valor: 25, Unidad: [Grados centígrados]

```
R : Registro := (To_Unbounded_String("13:00 pm"), 25, To_Unbounded_String("Grados centigrados"));
```

3. Imprimir por pantalla la variable definida anteriormente.

```
begin  
  
    Put_Line(To_String(r.hora));  
    Put_Line(Integer'image(r.valor));  
    Put_Line(To_String(r.unidad));  
  
end ejercicio6;
```

Resultado:

13:00 pm

25

Grados centígrados

3.7. Ejercicio 7 – TIPOS DINÁMICOS

Escribir un programa en ADA con las siguientes características:

1. Definir un tipo de datos “Nodo” para elaborar una lista enlazada de elementos tipo “Registro” definido en el ejercicio anterior.

```
type Registro is
  record
    hora : Unbounded_String := To_Unbounded_String("00:00 am");
    valor : integer := 1;
    unidad: Unbounded_String := To_Unbounded_String("");
  end record;
```

2. Inicializar una variable de tipo “Nodo” con tus datos personales.

```
type Nodo;
type Enlace is access Nodo;
type Nodo is
  record
    Valor : registro;
    Siguiente : Enlace;
  end record;
```

3. Imprimir por pantalla la variable definida anteriormente.

```
e : Enlace;
r : registro := (To_Unbounded_String("13:00 pm"), 25, To_Unbounded_String("Grados centígrados"));

begin

  e := new Nodo;
  e.Valor:= r;

  Put_Line(To_String(r.hora));
  Put_Line(Integer'image(r.valor));
  Put_Line(To_String (r.unidad));

end ejercicio7;
```

Resultado:

13:00 pm

25

Grados centígrados