

Práctico 10 - Enumerados, Conjuntos y Registros

Programación 1 InCo - Facultad de Ingeniería, Udelar

1. Determine cuáles de las siguientes declaraciones son válidas y explique por qué las restantes no lo son.

- | | |
|------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| <input type="checkbox"/> type Letra = ('X', 'Y', 'Z') | <input type="checkbox"/> procedure encontrar (var ciudad : (Minas, Florida, Flores)) |
| <input checked="" type="checkbox"/> type Lenguaje = (Pascal, Fortran, Basic) | <input checked="" type="checkbox"/> type Trabajo = (obrero, oficinista); |
| <input type="checkbox"/> type Codigo = (1, 2, 3, 4, 5) | ... |
| <input checked="" type="checkbox"/> type Codigo = (c1, c2, c3, c4, c5) | procedure buscar (var empleo : Trabajo); |

2. Dadas las siguientes declaraciones, determine cuáles de los siguientes fragmentos de código son válidos y explique por qué los restantes no lo son.

```
type color = (rojo, blanco, azul, purpura);  
var coloracion : color;
```

- | | |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|
| <input type="checkbox"/> read (rojo);
write (rojo) | <input type="checkbox"/> read (coloracion);
write (coloracion) |
| <input checked="" type="checkbox"/> coloracion := blanco;
case coloracion of
rojo : write ('rojo');
blanco : write ('blanco');
azul : write ('azul');
purpura: write ('purpura')
end | <input type="checkbox"/> coloracion := blanco;
write (coloracion) |
| | <input checked="" type="checkbox"/> if coloracion = azul then
write ('azul')
else
write ('no azul') |

3. Indique qué sucederá al ejecutar el siguiente programa.

```
program Ejercicio3;  
type Asignatura = (matematica, historia, computacion, geografia, fisica);  
var a, b: Asignatura;  
begin  
    a := matematica;  
    b := computacion;  
    if a > b then  
        write ('Magnifico')  
    else  
        write ('Excelente')  
end.
```

Excelente

4. Dadas las siguientes declaraciones:

```
type TipoDia = (lunes, martes, miercoles, jueves, viernes, sabado, domingo);  
var dia : TipoDia;  
    laborable : lunes..viernes;  
    finsemana : sabado..domingo;
```

Indique cuáles de las siguientes asignaciones producirán error de rango y explique por qué:


```

        bisiestro : (si, no)
    end;
var tiempo : Fecha;

```

Determine cuáles de las siguientes asignaciones son válidas y explique por qué las restantes no lo son:

- | | |
|--------------------------------------------------------|--------------------------------------------------------------|
| <input checked="" type="checkbox"/> tiempo.mes := 12 | <input type="checkbox"/> tiempo.bisiesto := 1 |
| <input type="checkbox"/> tiempo.fecha := 3-15-1900 | <input checked="" type="checkbox"/> tiempo.dia := tiempo.mes |
| <input type="checkbox"/> tiempo.fecha := mes | <input type="checkbox"/> tiempo.tiempo := tiempo |
| <input checked="" type="checkbox"/> tiempo.ano := 2002 | |

11. Los números complejos tienen dos componentes, una parte real y una parte imaginaria. Cada una de las componentes se representa mediante un número real. La siguiente declaración permite representar un número complejo:

```

type
    Complejo = record
        re, im : Real;
    end;

```

- (a) Escriba el procedimiento `sumaComplejos` que almacena en `c3` la suma de los números complejos `c1` y `c2`.

```

procedure sumaComplejos (c1, c2 : Complejo; VAR c3 : Complejo);

```

Si `re1` e `im1` representan los componentes de `c1` y `re2` e `im2` representan los componentes de `c2`, entonces los componentes de `c3` están dados por:

```

re3 = re1 + re2
im3 = im1 + im2

```

- (b) Escriba el procedimiento `multComplejos` que almacena en `c3` la multiplicación de los números complejos `c1` y `c2`.

```

procedure multComplejos (c1, c2 : Complejo; var c3 : Complejo);

```

Si `re1` e `im1` representan los componentes de `c1` y `re2` e `im2` representan los componentes de `c2`, entonces los componentes de `c3` están dados por:

```

re3 = re1 * re2 - im1 * im2
im3 = im1 * re2 + im2 * re1

```

- (c) Escriba un programa principal que lea dos números complejos y exhiba el resultado de su suma y multiplicación. Puede declarar subprogramas auxiliares que le permita cargar e imprimir un número complejo.

```

program complejos;

type
    Complejo = record
        re, im : real;
    end;
var
    c1, c2, resu : Complejo;

{Parte a}
procedure sumaComplejos (c1, c2 : Complejo; var c3 : Complejo);
begin
    c3.re:= c1.re + c2.re;
    c3.im:= c1.im + c2.im
end;

```

```

{Parte b}
procedure multComplejos (c1, c2 : Complejo; var c3 : Complejo);
begin
    c3.re := c1.re * c2.re - c1.im * c2.im;
    c3.im := c1.re * c2.im + c1.im * c2.re
end;

(* Procedimiento auxiliar que carga un numero complejo de la entrada *)
procedure cargar (var numero: Complejo);
begin
    write('Ingrese la parte real: ');
    readln(numero.re);
    write('Ingrese la parte imaginaria: ');
    readln(numero.im)
end;

(* Procedimiento auxiliar que muestra en pantalla un numero complejo *)
procedure mostrar (var numero: Complejo);
begin
    write (numero.re:8:2);
    if (numero.im > 0) then
        write (' + ', numero.im:8:2, 'i')
    else
        if (numero.im < 0) then
            write (' - ', abs(numero.im):8:2, 'i');
        writeLn
    end;

begin
    writeLn('Se va a cargar un número complejo');
    cargar(c1);
    writeLn('Se va a cargar otro número complejo');
    cargar(c2);
    suma(c1,c2,resu);
    write('La suma es: ');
    mostrar(resu);
    writeLn;
    multiplicacion(c1,c2,resu);
    write('El producto es: ');
    mostrar(resu);
    writeLn
end.

```

12. Se considera una versión simplificada de la wikipedia que contiene datos de artículos. Se asume que la wikipedia posee CANT_ARTICULOS artículos, y que el nombre de los artículos tiene exactamente CANT_LETRAS letras. Los idiomas de los artículos pueden ser inglés, portugués y español.

Para representar esta realidad se definen las siguientes declaraciones:

```

const
    CANT_LETRAS = ...; { valor entero mayor a 0 }
    CANT_ARTICULOS = ...; { valor entero mayor a 0 }

type
    TI idioma = (es, en, pt);

    TFecha = record
        dia : 1..31;
        mes : 1..12;

```

```

    anio : 2001..9999 (* La wikipedia comienza en 2001 *)
end;

TNombre = array [1..CANT_LETRAS] of char;

TArticulo = record
    nombre : TNombre;
    idioma : TIdioma;
    visitas : Integer;
    ultima_act : TFecha;
end;

Wikipedia = array [1..CANT_ARTICULOS] OF TArticulo;

```

- (a) Implemente la función `esPosterior` tal que dadas dos fechas `f1` y `f2`, devuelve `TRUE` si la fecha `f1` es posterior que la fecha `f2` y `FALSE` en caso contrario.

```
function esPosterior (f1, f2: TFecha) : boolean
```

```

function esPosterior (f1, f2: TFecha) : boolean;
begin
    EsPosterior := (f1.anio > f2.anio) or
        (f1.anio = f2.anio) and (f1.mes > f2.mes) or
        (f1.anio = f2.anio) and (f1.mes = f2.mes) and (f1.dia > f2.dia)
end;

```

- (b) Teniendo en cuenta que cada artículo tiene la fecha de su última actualización, implemente el procedimiento `articuloMasReciente` tal que dados la wikipedia y un idioma, devuelve el artículo que tiene la fecha más reciente en el idioma especificado. Asuma que en la wikipedia hay al menos un artículo en el idioma especificado.

```
procedure articuloMasReciente (wiki: Wikipedia; idioma: TIdioma; VAR art: TArticulo)
```

```

procedure articuloMasReciente (wiki: Wikipedia; idioma: TIdioma;
                               var art: TArticulo);
var i, j : integer;
begin
    i := 1;
    (* busco el primer articulo del idioma dado. Por el enunciado, se que siempre
    hay al menos uno *)
    while (wiki[i].idioma <> idioma) do
        i := i + 1;
    art := wiki[i];

    (* determino el articulo del idioma dado con mayor fecha de actualizacion *)
    for j := i + 1 to CANT_ARTICULOS do
        if (wiki[j].idioma = idioma) and esPosterior(wiki[j].ultima_act,
            art.ultima_act) then
            art := wiki[j];
    end;
end;

```

- (c) Implemente el procedimiento `imprimirArticulosMasRecientes` tal que para cada idioma imprime el nombre del artículo más reciente en dicho idioma, junto con su cantidad de visitas y la fecha de su última actualización. Asuma que en la wikipedia hay al menos hay un articulo en cada uno de los tres idiomas.

```
procedure imprimirArticulosMasRecientes (wiki: Wikipedia)
```

```

(* Procedimiento auxiliar que imprime un articulo *)
procedure imprimirArticulo (a: TArticulo);
var i : integer;
begin
  (* Imprimo nombre *)
  write ('Nombre: ');
  for i := 1 to CANT_LETRAS do
    Write (a.nombre[i]);
  writeLn;

  (* Imprimo idioma *)
  write ('Idioma: ');
  case a.idioma of
    pt: writeLn('Portugues');
    en: writeLn('Ingles');
    es: writeLn('Español');
  end;

  (* Imprimo visitas *)
  writeLn ('Visitas: ', a.visitas);

  (* Imprimo ultima_act *)
  writeLn ('Ultima actualizacion: ', a.ultima_act.dia:2, '/',
    a.ultima_act.mes:2, '/', a.ultima_act.anio:4);
end;

procedure imprimirArticulosMasRecientes (wiki: Wikipedia);
const PRIMER_IDIOMA = es;
      ULTIMO_IDIOMA = pt;
      CANT_IDIOMAS = 3;
var arts : Array [TIdioma] of TArticulo;
    encuentre : set of TIdioma;
    i, j : integer;
    id : TIdioma;
begin
  i := 1;
  j := 0;
  encuentre := [];
  repeat
    if wiki[i].idioma in encuentre then
      begin
        if esPosterior(wiki[i].ultima_act, arts[wiki[i].idioma].ultima_act) then
          arts[wiki[i].idioma] := wiki[i]
        end else
          begin
            encuentre := encuentre + [wiki[i].idioma];
            arts[wiki[i].idioma] := wiki[i];
            j := j + 1;
          end;
        i := i + 1;
      end
    until j = CANT_IDIOMAS;
    for j := i to CANT_ARTICULOS do
      if esPosterior(wiki[j].ultima_act, arts[wiki[j].idioma].ultima_act) then
        arts[wiki[j].idioma] := wiki[j];
      for id := PRIMER_IDIOMA to ULTIMO_IDIOMA do
        imprimirArticulo(arts[id])
      end;
    end;
end;

```

13. Se considera una versión simplificada para la gestión de la información de los salones que integran una

facultad. Se asume que la facultad posee CANT_SALONES salones, y que cada salón tiene un máximo de MAX_PIZARRONES pizarrones.

Para representar esta realidad se definen las siguientes declaraciones:

```
const
  CANT_SALONES = ...; { valor entero mayor a 0 }
  MAX_PIZARRONES = ...; { valor entero mayor a 0 }

type
  TSalon = record
    asientos : Integer;
    pizarrones : 1..MAX_PIZARRONES;
    hayProyector : Boolean;
  end;

  TFacultad = array [1..CANT_SALONES] of TSalon;
```

- (a) Implemente el procedimiento `informeSalones` tal que imprima un informe de todos los salones de la facultad, incluyendo, por cada uno de ellos, cantidad de asientos, pizarrones, y si tiene proyector o no.

```
procedure informeSalones (facu: TFacultad);
```

```
procedure informeSalones (facu: TFacultad);
var i : 1 .. CANT_SALONES;
begin
  for i := 1 to CANT_SALONES do
  begin
    write('El salón ',i:1,' tiene ',facu[i].asientos:1,' asientos,',
          facu[i].pizarrones:1,' pizarrones y ');
    if facu[i].hayProyector then
      writeLn('tiene proyector.')
    else
      writeLn('no tiene proyector.')
    end
  end
end;
```

- (b) Implemente el procedimiento `salonMasAsientos` tal que devuelve el índice de la celda correspondiente al salón con la mayor cantidad de asientos de la facultad junto con la cantidad de asientos correspondiente. En caso de haber dos o más salones con la mayor cantidad de asientos, devuelve el primero de ellos.

```
procedure salonMasAsientos (facu: TFacultad; var indSalon: Integer;
                           var maxAsientos: Integer);
```

```
procedure salonMasAsientos (facu: TFacultad; var indSalon: Integer;
                           var maxAsientos: Integer);
var i : 1 .. CANT_SALONES;
begin
  indSalon := 1;
  for i := 2 to CANT_SALONES do
    if facu[i].asientos > facu[indSalon].asientos then
      indSalon := i;
      maxAsientos := facu[indSalon].asientos
    end
  end;
```

- (c) Implemente la función `primerSalonSinAsientos` tal que devuelve el índice de la celda correspondiente al primer salón de la facultad que no tiene ningún asiento. En caso de que no haya ningún salón sin asientos, devuelve cero. Su encabezado es el siguiente:


```
function primerSalonSinAsientos (facu: TFacultad) : Integer;
```

```
function primerSalonSinAsientos (facu: TFacultad) : Integer;  
var i : integer;  
begin  
  i := 1;  
  while (i <= CANT_SALONES) and (facu[i].asientos <> 0) do  
    i := i + 1;  
  if i <= CANT_SALONES then  
    primerSalonSinAsientos := i  
  else  
    primerSalonSinAsientos := 0  
end;
```