

Enumerado, subrango y registro

Introducción a la programación II

Tipos de datos en Pascal

En Pascal los tipos pueden ser:

- Tipos simples:

Tipos primitivos:

entero (integer), real (real), carácter (char), lógico (boolean).

Definidos por el usuario:

enumerado y **subrango**.

- Tipos estructurados:

Arreglos (array), **registros** (record), archivos (file), conjuntos (set).

- Tipo puntero

Enumerado

Sirven para crear nuevos tipos ordinales como lo son los números, caracteres, y true/false.

Para definir un tipo enumerado, se listan los identificadores que pueden tomar los datos de este tipo.
Declaración:

type

```
nombre_tipo = (constante1,constante2,...,constanteN);  
//constante1,constante2,...,constanteN son ordinales definidos por el usuario  
//o puede utilizarse otros ordinales de Pascal
```

Ejemplo:

type

```
dias_semana = (lunes,martes,miercoles,jueves,viernes,sabado,domingo);
```

var

```
dias : dias_semana;
```

El propósito principal de los tipos enumerados es permitir al programador el uso de nombres con significado para los datos que necesita utilizar.

Observación: No pueden utilizarse directamente por los procedimientos de entrada/salida, como por ejemplo read(dias). En esos casos se necesita una conversión con una variable intermedia.

Enumerado: criterio de orden y operaciones

Teniendo en cuenta de izquierda a derecha los nombre declarados como ordinales, se mantiene entre si un criterio de orden:

```
type      nombre_tipo = (constante1,constante2,...,constanteN);  
           //Orden      0          1          N  
var      variable_enumerado : nombre_tipo ;
```

De acuerdo al orden de enumeración quedan definidas las siguientes operaciones:

Comparaciones: = <> < > <= >=

Acceso a los contiguos: **succ(variable_enumerado); //obtiene el siguiente valor enumerado**
 pred(variable_enumerado); //obtiene el anterior valor enumerado

Conversión de enumerado a entero:
 ord(variable_enumerado); //obtiene el orden del valor enumerado

Conversión de entero a enumerado:
 Tipo_ordinal(orden enumeración); //obtiene el valor enumerado

Enumerado: criterio de orden y operaciones

Ejemplo:

```
type    dias_semana = (lunes,martes,miercoles,jueves,viernes,sabado,domingo);  
        //Orden      0      1      2      3      4      5      6  
var     dia : dias_semana;
```

For **dia:=lunes** to **domingo** do

...

dia:=lunes;

While **dia <= domingo** do begin

...

dia:=succ(dia);

end;

...

dia:=lunes;

variable_entero:=ord(dia); //variable_entero=0

...

...

dia:=dias_semana (1); //dia=martes

...

Ejemplo moodle

//Ejemplo: hacer un programa que declare un tipo enumerado para los días de la semana.
//Luego mediante una variable de dicho tipo recorra los días de la semana e imprima en cada
//caso a que día corresponde.

```
program dias_semana;  
type  
    dia_semana = (lunes,martes,miercoles,jueves,viernes,sabado,domingo);  
  
var  
    dias : dia_semana;  
  
begin  
    for dias:=lunes to domingo do  
        case dias of  
            lunes: writeln('lunes');  
            martes: writeln('martes');  
            miercoles: writeln('miercoles');  
            jueves: writeln('jueves');  
            viernes: writeln('viernes');  
            sabado: writeln('sabado');  
            domingo: writeln('domingo')  
        end;  
  
end.
```

Ejemplo moodle

//Ejemplo: hacer un programa que declare un tipo enumerado para los días de la semana.
//Luego solicite ingresa por teclado un nro de día e imprima al día que corresponde.

```
program dias_semana;  
type  
    dia_semana = (lunes,martes,miercoles,jueves,viernes,sabado,domingo);  
  
var  
    dia : dia_semana;  
    nro_dia:integer;  
  
begin  
    writeln('ingrese 0 para el lunes, 1 para martes..., 6 para domingo');  
    readln(nro_dia);  
    dia:=dia_semana(nro_dia);  
    case dia of  
        lunes: writeln('lunes');  
        martes: writeln('martes');  
        miercoles: writeln('miercoles');  
        jueves: writeln('jueves');  
        viernes: writeln('viernes');  
        sabado: writeln('sabado');  
        domingo: writeln('domingo')  
    end;  
  
end.
```

Subrango

Un tipo subrango se define a partir de un tipo ordinal, especificando dos valores ordinales de ese tipo que actúan como límite inferior y superior del conjunto de datos.

El tipo subrango es un tipo ordinal y sus valores se ordenan de igual modo que en el tipo del que se deducen.

Declaración:

type

nombre = limite inferior .. limite superior;

Ejemplos:

type

digito = 0 .. 9;

Consta de los elementos
0,1,2,3,4,5,6,7,8,9

type

digito_caracter = '0' .. '9';

Consta de los caracteres '0' a '9'

Ejemplo moodle

//Ejemplo: hacer un programa que declare un tipo enumerado y subrangos de días de trabajo y de descanso
//Luego mediante variables de cada tipo recorra los días que incluyen en cada caso.

program dias_semana;

type

dia_semana = (lunes,martes,miercoles,jueves,viernes,sabado,domingo);

dia_trabajo = lunes..viernes;

dia_descanso= sabato..domingo;

var

diat: dia_trabajo;

diad: dia_descanso;

begin

writeln('Dias de trabajo:');

for diat:=lunes to viernes do

case diat of

lunes:writeln('lunes ');

martes:writeln('martes ');

miercoles:writeln('miercoles');

jueves:writeln('jueves ');

viernes:writeln('viernes ');

end;

writeln('Dias de descanso:');

for diad:=sabado to domingo do


case diad of

sabado:writeln('sabado ');

domingo:writeln('domingo');

end;

end.



**Si no se definen los valores
posibles de los no
predefinidos, no es posible
definir un subrango
asociado**

Registro

- Es un tipo estructurado representado por un identificador, mediante el cual se pueden representar múltiples datos individuales, y donde cada uno de ellos puede ser referenciado de forma separada.
- **Representa un conjunto de datos no necesariamente del mismo tipo.**
- Los elementos individuales se llaman campos del registro.

Declaración:

```
type
    nombre_reg = record
        campo1;
        campo2;
        ...
        campoN;
    end;
```

Cada declaración de **campo** es similar al de una variable individual: **variable:tipo**

Registro

Ejemplo: cuenta de cliente con número y saldo.

```
type      cuenta=record
                        numcliente:integer;
                        saldocliente:real
end;
var       cliente1,cliente2:cuenta;
```



Un registro con dos campos

La forma de acceder a un elemento de un registro es: **nombre_de_variable.campo**

Ejemplo: **cliente1.numcliente**

Dos registros del mismo tipo pueden copiarse íntegramente mediante una asignación.

Ejemplo: **cliente1:=cliente2;**

Observación: dependiendo de la versión de pascal se puede requerir que la copia entre registros se realice campo a campo. Se recomienda siempre hacer este tipo de copia.

```
cliente1.numcliente:=cliente2.numcliente;
cliente1.saldocliente:=cliente2.saldocliente;
```

Registro

Ejemplo: cuentas de clientes con tarjetas, saldo, y fecha de pago.

```
const
    MAX=20;
type
    tarjeta= (visa,4b,clavecard);
    fecha=record
        mes:1..12;
        dia:1..31;
        anio:1900..2100
    end;
    cuenta=record
        nombrecliente:string;
        tarjetacliente:tarjeta;
        saldocliente:real;
        ultimopago:fecha
    end;
    clientes=array[1..MAX] of cuenta;
```

A diferencia de los días de la semana, no se definen los valores posibles porque son predefinidos

Accesos:

```
var clientepreferido:cuenta; arrcliente:clientes;
clientepreferido.saldocliente----->(es un real)
clientepreferido.nombrecliente----->(es un string)
arrcliente[3].tarjetacliente----->(de tipo tarjeta)
arrcliente[5].ultimopago.dia ----->(de tipo entero entre 1 y 31)
```

Ejemplo moodle

//Hacer un programa que defina un registro con datos de alumno, cargue una variable de este tipo
//desde el principal con valores, e imprima su contenido

program ejemplo_record;

type

datos_alumno=record

inicial_nombre:char;

inicial_primer_apellido:char;

inicial_segundo_apellido:char;

edad:integer;

end;

var

estudiante : datos_alumno;

begin

estudiante.inicial_nombre := 'G';

estudiante.inicial_primer_apellido := 'B' ;

estudiante.inicial_segundo_apellido := 'P';

estudiante.edad := 18;

writeln('El estudiante ', estudiante.inicial_nombre, estudiante.inicial_segundo_apellido,
estudiante.inicial_primer_apellido, ' tiene ', estudiante.edad, ' anios')

end.

Ejemplo moodle

//Hacer un programa que defina un arreglo de registros con
//datos de alumno, cargue el arreglo y lo imprima

program ejemplo_record;

const

MAX=3;

type

datos_alumno=record

nombre, apellido:string;

edad:integer;

end;

arreglo_alumnos=array[1..MAX] of datos_alumno;

procedure cargar_arr_alumnos(var ar:arreglo_alumnos);

var i:integer;

begin

for i:=1 to MAX do begin

writeln('Ingrese nombre: ');

readln(ar[i].nombre);

writeln('Ingrese apellido: ');

readln(ar[i].apellido);

writeln('Ingrese edad: ');

readln(ar[i].edad);

end;

end;

procedure imprimir_arr_alumnos(ar:arreglo_alumnos);

var i:integer;

begin

for i:=1 to MAX do

writeln('Nombre: ',ar[i].nombre,', Apellido: ',ar[i].apellido,',

Edad: ',ar[i].edad);

end;

var

ar : arreglo_alumnos;

begin

cargar_arr_alumnos(ar);

imprimir_arr_alumnos(ar);

end.

Ejemplo moodle

//Hacer un programa que defina un arreglo de registros con datos de alumno,
//cargue el arreglo, guarde, lea, y lo imprima

program ejemplo_record;

const

MAX=3;

type

datos_alumno=record

nombre, apellido:string;

edad:integer;

end;

arreglo_alumnos=array[1..MAX] of datos_alumno;

archivo_alumnos=file of arreglo_alumnos;

procedure cargar_arr_alumnos(var ar:arreglo_alumnos);

var i:integer;

begin

for i:=1 to MAX do begin

writeln('Ingrese nombre: ');

readln(ar[i].nombre);

writeln('Ingrese apellido: ');

readln(ar[i].apellido);

writeln('Ingrese edad: ');

readln(ar[i].edad);

end;

end;

procedure guardar_arr_alumnos(ar:arreglo_alumnos;nombre_archivo:string);

var arch:archivo_alumnos;

begin

assign(arch,nombre_archivo);

rewrite(arch);

write(arch,ar);

close(arch);

end;

procedure leer_arr_alumnos(var ar:arreglo_alumnos;nombre_archivo:string);

var arch:archivo_alumnos;

begin

assign(arch,nombre_archivo);

reset(arch);

read(arch,ar);

close(arch);

end;

procedure imprimir_arr_alumnos(ar:arreglo_alumnos);

var i:integer;

begin

for i:=1 to MAX do

writeln('Nombre: ',ar[i].nombre,', Apellido: ',ar[i].apellido,', Edad: ',ar[i].edad);

end;

var

ar : arreglo_alumnos;

begin

cargar_arr_alumnos(ar);

guardar_arr_alumnos(ar,'/ip2/archivo_alumnos.dat');

leer_arr_alumnos(ar,'/ip2/archivo_alumnos.dat');

imprimir_arr_alumnos(ar);

end.