UNIVERSIDAD SIMÓN BOLÍVAR
Departamento de Computación y Tecnología de la Información

CI2691: Laboratorio de Algoritmos y Estructuras I

TALLER 02: Tipos subrango y enumerados. Expresiones aritméticas y lógicas, precedencia. Asignación, instrucción nula y secuenciación. Indentación. Documentación del código. Escritura de aserciones intermedias como expresiones lógicas.

Tipos subrango y enumerados [1]

Sección "3.1.1 Ordinal types", sólo tipo "Enumetarion types" y tipo "Subrange types"

Ejercicio:

Defina un tipo en PASCAL para:

Los días de la semana de (lunes a domingo)

Los días del fin de semana (sábado a domingo) como enumerado independiente del anterior

Los días del fin de semana (sábado a domingo) como subrango de los días de la semana

Haga un programa en que se visualicen los ordinales de todos y cada uno de los elementos de los tipos tres tipos definidos anteriormente.

Expresiones aritméticas y lógicas, precedencia [1]

Capítulo "12. Expressions", secciones "12.1 Expression syntax", "12.8.1 Arithmetic operators", "12.8.1 Booelan operators"

Asignación, Instrucción nula, Secuenciación [1]

Capítulo "13 Statements", secciones "13.1.1 Assignments", sólo ":=", "13.2.1. Compound statments"

Ejercicio: Completar los programas del taller pasado.

Indentación [2]

La indentación se refiere al espacio en blanco dejado al comienzo de la línea con el propósito de hacer más legible el código.
Por lo general se agrega indentación cada vez que se abre un nuevo bloque o sección de código, por ejemplo: un programa, una lista de declaraciones de variables, una lista de instrucciones en secuencia dentro de una acción compuesta.
Los editores que reconocen la sintaxis del lenguaje de programación suelen ayudar a tener una buena identación.
Un ejemplo de indentación en instrucción compuesta es el siguiente:

```
begin

      readln(variableEntrada);
```

```
      write(expresionSalida)
end.
```

Longitud de las Líneas

Se debe evitar tener líneas de más de 80 carácteres, dado que por lo general éstas tienden a hacer la lectura del programa complicada y al imprimir el código se dificulta mucho su lectura.

Cómo cortar Líneas#

Cuando una expresión no cabe en una línea se debe partir tomando en cuenta los siguientes principios:

- Cortar luego de una coma.
- Cortar antes de un operador.
- La nueva línea debe estar indentada al mismo nivel en donde comienza la expresión en la línea anterior.
- Es preferible cortar en la expresión más externa.

Si al aplicar las reglas anteriores el código obtenido es confuso, o está muy ajustado a la derecha, entonces use en su lugar una indentación de ocho espacios.

A continuación se presentan algunos ejemplos de como cortar líneas en llamadas a procedimientos:

```
writeln(algunaExpresionLarga1, algunaExpresionLarga2, algunaExpresionLarga3,

        algunaExpresionLarga4, algunaExpresionLarga5);
```

Los siguientes son dos ejemplos de cortar expresiones aritméticas. El primero es preferible, dado que el corte ocurre afuera de la expresión parentizada.

```
nombreLargo1 := nombreLargo2 * (nombreLargo3 + nombreLargo4 - nombreLargo5)

                + 4 * nombreLargo6; //PREFERIBLE



nombreLargo1 := nombreLargo2 * (nombreLargo3 + nombreLargo4

                               - nombreLargo5) + 4 * nombreLargo6; //EVITAR
```

Documentación del código [2]

Los comentarios en PASCAL son delimitados por (*...*) , {}. y //.

Los comentarios deben ser usados para dar una visión del código y proporcionar información adicional que no se encuentra disponible en el mismo. Los comentarios sólo deben contener información necesaria para leer y enteder el programa. Por ejemplo información sobre como se debe compilar o ejecutar el programa o en que directorio se encuentra no debe ser incluida como comentario.

Se debe incluir información acerca decisiones de diseño del programa que no sean triviales o que no sean obvias al leer del código. Debe tenerse el cuidado de no repetir información que ya esté presente en el código y pueda ser obtenida claramente a partir de éste. Los comentarios redundantes por lo general quedan obsoletos con el tiempo. En general, evite cualquier comentario que pueda quedarse obsoleto mientras el código evoluciona.

Nota: La frecuencia de los comentarios por lo general refleja poca calidad de código. Cuando se sienta forzado a agregar un comentario, considere reescribir el código para hacerlo más claro.

Los comentarios no deben incluirse en cajas formadas con asteriscos y otros caracteres. Los comentarios no deben incluir caracteres especiales

Adoptaremos como regla de estilo el colocar comentario en

En el encabezado del programa, comentario de bloque entre (* ... *)

En la declaración de cada variable o constante, comentario de cola iniciado con ///

Para delimitar secciones del código, comentarios de línea entre (* .. *)

Para colocar precondiciones, postcondiciones y aserciones intermedias entre { .. }

Escritura de aserciones intermedias como expresiones lógicas

Usaremos las mismas convenciones del lenguaje Gacela para escribir precondiciones, postcondiciones y aserciones intermedias

Adicionalmente, necesitamos saber escribirlas en PASCAL usando los operadores aritméticos y booleanos. Como aún no sabemos chequear condiciones booleanas en PASCAL, a efectos de este laboratorio colocaremos ambas como comentarios entre { .. }.

Note que: Las aserciones expresadas con cuantificadores o funciones de agregación que podemos escribir en lógica, pueden también escribirse con la nomenclatura de Gacela no las podemos escribir en PASCAL.

Ejercicio: Diseñe e implemente un programa en PASCAL para cada uno de los siguientes problemas:

Dado el largo y ancho de una habitación mostrar su superficie con cuatro decimales.

Dado el radio de una circunferencia (número real), y calcule e imprima el área, el perímetro de la circunferencia, y el volumen de la esfera asociada. Use 3.1416 como valor de π .

Dada cierta cantidad de tiempo en segundos introducida por el usuario, desglósela en su equivalente en semanas, días, horas, minutos y segundos.

Dadas dos variables enteras A y B, intercambiar sus valores.

Dado un número natural n, calcular $\sum_{i=0}^{n} i^2$

Referencias

1. *Michaël Van Canney Free Pascal : Reference guide. Reference guide for Free Pascal, version 2.6.0, Document version 2.6, December 2011, Disponible en la web:* http://www.freepascal.org/docs.var
2. *Gabriela Montoya, Jorge Guerra, Víctor Fuentes,Wendi Urribarrí, Yolifé Arvelo, y Jesús Ravelo Guía de Estilo Para Programas en GaCeLa, Disponible en la web:* http://wiki.lal.labf.usb.ve/GacelaWiki/Wiki.jsp?page=GuiaDeEstilo1-4
3. *Damaris Candanedo y Jorge Guerra, Guia para escribir programas en GaCeLa .Última actualización: Mayo 28, 2003, Disponible en la web:* http://ldc.usb.ve/~gpalma/cursos/ci2691/guiagacela.html