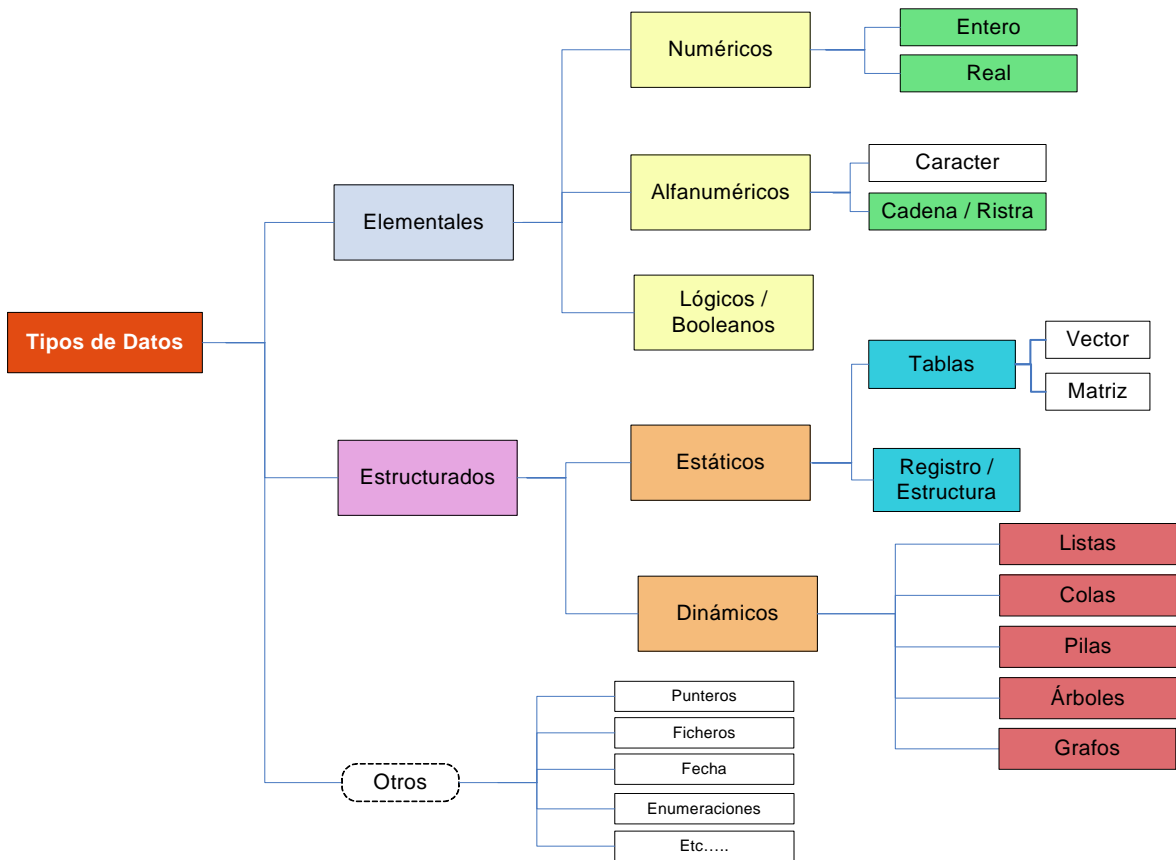


UD 2: Tipos de Datos

Tipos de datos



Pseudocódigo → Declaración de constantes

Constante real PI = 3.14159265
 enteras duplica = 2, divide = 4

Podemos observar el uso del singular o plural según convenga

Pseudocódigo → Declaración de tipos

tipo nombre_tipo es tipo_dato {rangoini .. rangofin} fin tipo

tipo nombre_tipo es estructura
campo TipoDato nombre

...

fin tipo

Ejemplo

```

tipo nodo es estructura
  campo t_dato dato
  campos puntero a nodo izq, der
fin tipo
    
```

Pseudocódigo → Declaración de variables

Variable entera Var1

Variable real Var2

Variable ristra Var3

Variable carácter Var4

Variable lógica Var5

Variable puntero a *TipoDato* Var9

Variable *TipoDato* Var7[num]

Variable *TipoDato* Var8[num][num]

Normas para el nombre de las variables → *Identificadores*

- Alfanuméricos (combinación de letras y números)
- Tienen que comenzar por una letra
- SIN espacios
- Deben tener lógica el nombre con el uso/información
- El tamaño del nombre depende del compilador.

Tipo de una variable/dato

- Establece el rango o intervalo de valores que puede usar.
- Determina el espacio de memoria que se reservará para el dato.

*** Tipo Numérico Entero

- Para representar números enteros (sin cifras decimales)
- Pueden llevar o no el signo + o el –

*** Tipo Numérico Real

- Para representar números reales (con cifras decimales)
- Para representar números o muy grandes o muy pequeños

*** Tipo Caracter

- Para representar un carácter de tipo alfanumérico
- El carácter se encerrará entre comillas simples: '5', 'a', '/'

*** Tipo Cadena/Ristra

- Para representar un conjunto de caracteres alfanuméricos
- Se encerrarán entre comillas dobles: "Hola alumn@s"
- La cadena vacía es "".
- Hay que tener en cuenta que al comparar ristras, se comprueba carácter a carácter, y no por el tamaño de la misma, así mismo, hay que tener en cuenta que las letras minúsculas se consideran mayores a las mayúsculas, y los números menores de las letras. Esto viene por el valor ASCII de cada letra.
- En la definición se puede especificar el tamaño:
 - **Variable ristra** var1[num]
- Para el manejo de cadenas tenemos tres funciones importantes
 - **lon**(cadena) → Devuelve la longitud de la cadena
 - **pos**(cadena_origen, subcadena) → Busca la *subcadena* en *cadena_origen* y devuelve la posición de inicio de ésta. Siempre empieza por el principio de *cadena_origen*.
 - **sub**(cadena, posinicio, tamaño) → Devuelve una subcadena de longitud *tamaño* en *cadena* a partir de la posición indicada en *posinicio*.
 - La concatenación se realizará con el +

*** Tipo Lógico

- Para representar dos valores especiales: Verdadero / Falso
- No pueden tomar otro valor que el indicado anteriormente.

*** Tipo Puntero

- Para contener la dirección de memoria de otra variable
- De gran utilidad para operaciones con estructuras dinámicas
- Funciones de gran utilidad en asignación dinámica es
 - Variable ← **TomarBloque**(tipo)
 - **Liberar**(variable)

*** Tipo Ficheros

- Existen tres tipos de ficheros fundamentales:
 - **FicheroTexto** → Ficheros de textos / Acceso secuencial
 - **Fichero** → Ficheros no uniformes / Acceso secuencial
 - **Fichero<Tipo>** → Ficheros uniformes / Acceso directo
- Una vez declarados deberemos abrirlos:
 - **Abrir(Fich, NombreFich, Lectura | Escritura | Lectura/Escritura)**
Devolverá verdadero o falso según se realice la operación con éxito
- Si el fichero físico no existe deberemos crearlo
 - **CrearFichero(NombreFich)**
- Para manejar el fichero tenemos las siguientes funciones
 - **Leer (Fich, variable)**
 - **Escribir (Fich, variable)**
 - **FinFichero(Finch)**
 - **Tamaño(Fich)** → Devuelve el tamaño en bytes excepto para los de acceso directo que devuelve el número de componentes.
 - **IrA(Fich, Pos)**
 - **Posición (Fich)** → Devuelve la posición actual en bytes o registros.
 - **EscribirLínea(Fich, var)** → Para ficheros de textos
 - **LeerLínea (Fich, var)** → Para ficheros de textos
 - **FinLínea (Fich)** → Para ficheros de textos
 - **SaltarLínea (Fich)** → Para ficheros de textos
 - **NuevaLínea (Fich)** → Para ficheros de textos
- Otras funciones
 - **Renombrar (NombreViejo, NombreNuevo)**
 - **Eliminar (NombreFich)**

Operadores

<u>Operador</u>	<u>Símbolo</u>	<u>Significado</u>
Paréntesis	()	
Aritméticos	** , ^ * / div, \ %, mod + -	Potencia Producto División División entera Módulo (resto de la división) Signo positivo o suma Signo negativo o resta
Alfanuméricos	+	Concatenación
Relacionales	== , = !=, <> < <= > >=	Igual a Distinto a Menor que Menor o igual que Mayor que Mayor o igual que
Lógicos	!, NOT, no &&, AND, y, ∧ , OR, o, ∨	Negación Conjunción Disyunción

Tipos de Instrucciones

Instrucciones Primitivas

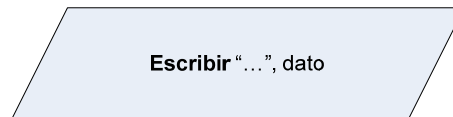
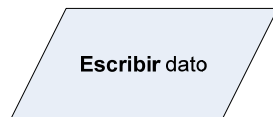
Instrucciones de Entrada

Encargadas de leer/recoger el dato de un periférico de entrada



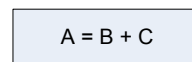
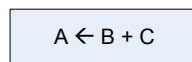
Instrucciones de Salida

Instrucciones cuyo cometido es mostrar un dato/valor en un dispositivo de salida.



Instrucciones de Asignación

Instrucciones cuyo cometido es almacenar un dato o un valor en una variable



Instrucciones Compuestas

Instrucciones que no pueden ser ejecutadas directamente por el procesador, y están constituidas por un bloque de acciones agrupadas en subrutinas, subprogramas, funciones o módulos.



Instrucciones de control

Instrucciones de Salto

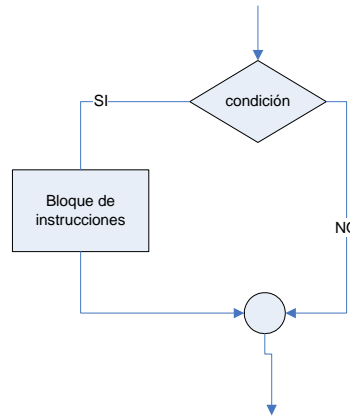
No se usarán. No usaremos el GOTO.

Instrucciones Condicionales

si {condición} **entonces**

.....

fin si



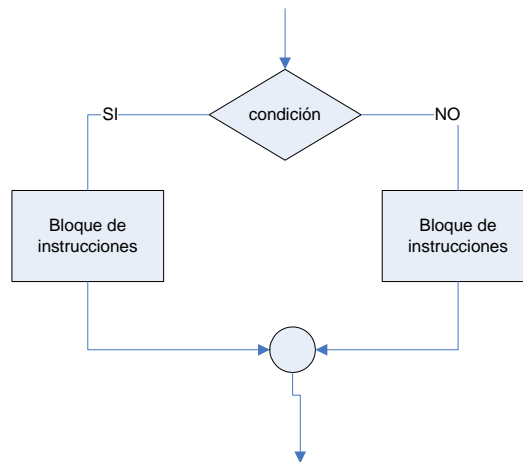
si {condición} **entonces**

.....

si no

.....

fin si



según {expresión} **hacer**

Valor1:

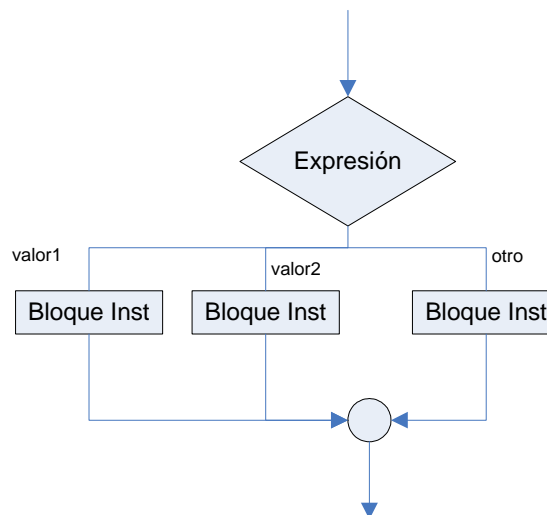
Valor2:

...

ValorN:

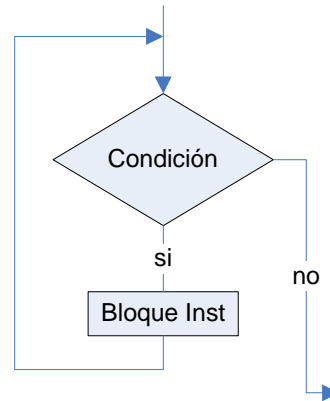
otro:

fin según

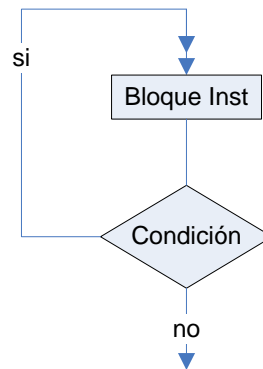


Instrucciones Repetitivas

mientras {condición} **hacer**
.....
fin mientras

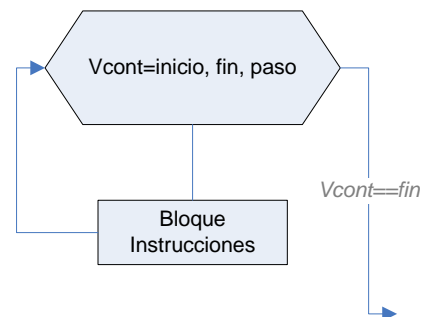


repetir
.....
mientras {condición}



repetir
.....
hasta {condición}

para VariableContadora **de** Inicio **a** Fin [**paso** valor] **hacer**
.....
fin para



Ejercicios propuestos

- 1) Diseñar un algoritmo que calcule una ecuación de 2º grado ($ax^2+bx+c=0$).
- 2) Diseño de un algoritmo que sume los 100 primeros números introducidos por teclado.
- 3) Diseño de un algoritmo que tras leer N números muestre el mayor y el menor de ellos.
- 4) Programa que escriba la tabla de multiplicar de un número introducido por teclado.
- 5) Que tipo de datos resulta más adecuado para representar cada uno de los conceptos siguientes :
 - a) El sueldo de un trabajador
 - b) La edad de una persona
 - c) El número de hijos
 - d) El estado civil
 - e) El estado de caducado no de un producto
 - f) El nº de teléfono
 - g) La dirección.
- 6) Escribir un programa que pida una cantidad en pesetas y la convierta en euros.
- 7) A partir del programa anterior escribir uno que pase de pesetas a Libras esterlinas, pidiendo primero cuantas pesetas es una libra.
- 8) Para cada uno de los puntos siguientes escribir un programa que pida los datos necesarios y calcule el área y el perímetro de la figura indicada
 - a) Un cuadrado
 - b) Un rectángulo
 - c) Un triángulo
 - d) Un círculo

Programación en Lenguajes Estructurados

- 9) Escribir un programa para calcular el importe de una venta en un supermercado. El usuario debe indicar el nombre del producto, el precio por unidad y el nº de unidades y el programa sacará por pantalla el nombre del producto, el nº de unidades vendidas y el precio total. Preste especial atención a qué tipo de datos resulta más adecuado para cada representar cada cantidad.
- 10) Escribir un programa que calcule la nómina de un trabajador de la manera siguiente. El trabajador cobra un precio fijo por hora y se le retiene un 5% en concepto de IRPF. El programa debe pedir el nombre del trabajador, las horas trabajadas y el precio que cobra por hora. Como salida debe imprimir el sueldo bruto, la retención y el sueldo neto.
- 11) Escribir un programa que pida un número entero y saque por pantalla el cociente y el resto de la división entera entre ambos
- 12) Escribir un programa que pida una hora en segundos y la saque por pantalla en el formato "hh:mm:ss", es decir horas, minutos y segundos
- 13) Escribir un programa que pida un número entero y determine si es múltiplo de 2 y de 5
- 14) Escribir un programa que pida la nota de un examen (un nº real entre 0 y 10) e imprima por pantalla la calificación en formato "Suspenso", si la nota es menor que 5, "Aprobado" si está entre 5 inclusive y 7 sin incluir, "Notable" si está entre 7 inclusive y 9 sin incluir, "Sobresaliente" si está entre 9 inclusive y 10 sin incluir y "Matrícula de honor" si la nota es igual a 10.

Bibliografía

- **Programación en Lenguajes Estructurados** (Desarrollo de Aplicaciones Informáticas). *Enrique Quero Catalinas*. Editorial Paraninfo. 2001.
- **Enciclopedia Libre Wikipedia** – <http://es.wikipedia.org>