

## Unidad III Estructura general de un programa

M.C. Juan Carlos Olivares Rojas

## Agenda

- 3.1 Concepto de variable y constante
- 3.2 Expresiones
- 3.3 El ciclo precondicional
- 3.4 El ciclo postcondicional
- 3.5 Variables contadoras y acumuladoras
- 3.6 Módulos, funciones y procedimientos de los algoritmos

### 3.1 Concepto de variable y constante

- Variable: tipo identificador
- Constante: tipo identificador de que es constante

#### DECLARACIONES

A: entero

B: logico

#### INICIO

A <- 1.3 x

A <- B x

A <- "UNID" x

#### FIN

#### DECLARACIONES

A: entero

Pi: 3.1416

#### INICIO

A <- 10

Pi <- 10 x

#### FIN

### 3.2 Expresiones

- Las expresiones pueden ser aritméticas, de asignación y condicionales.
- Las expresiones aritméticas generalmente están asociadas a valores numéricos. Si son de diferentes tipos numéricos se convierten al tipo del de mayor valor (flotantes, enteros con signo, enteros sin signo, enteros cortos)

### Expresiones

- Las expresiones de asignación también hace esa conversión de tipo entre valores numéricos. A esta conversión se le llama cast.
- Las expresiones condicionales se pueden evaluar en "corto circuito" buscando para las AND que exista un 0: el predicado es falso, y para el OR un 1: el predicado es cierto.

### 3.3 El ciclo precondicional

- Un ciclo es un conjunto de instrucciones que se repiten un determinado número de veces.
- El objetivo de la informática es la automatización de actividades repetitivas.
- Imagine que tenemos que hacer el desayuno para toda la familia, debemos repetir las instrucciones para hacer huevo n veces, donde n representa el número de miembros de la familia.

## Ciclos precondicionales

### HACER 4 VECES

Romper cascarón

Freír huevo

Servir huevo

Fin

- Existen dos tipos de ciclos: precondicionales y postcondicionales.



## Ciclos precondicionales

- Los ciclos precondicionales son aquellos en los que las condiciones lógicas se evalúan al inicio del conjunto de instrucciones a repetir.
- Existen dos tipos de ciclos precondicionales: MIENTRAS y DESDE\_HASTA.
- Los ciclos precondicionales garantizan que las instrucciones se ejecuten de 0 a n veces



## 3.4 El ciclo postcondicional

- Los ciclos condicionales son aquellos que evalúan la condición lógica al final del conjunto de instrucciones a repetir; por lo que nos garantiza que las instrucciones se repitan al menos una vez.
- El único ciclo postcondicional es el HACER\_MIENTRAS



## 3.5 Variables contadoras y acumuladoras

- Las variables contadoras sirven para llevar el control del número de iteraciones que un grupo de instrucciones se ha ejecutado. Generalmente son variables del tipo entero
- Las variables acumuladoras sirven para conservar el resultado de varios procesos que se realizan dentro de un ciclo. Pueden ser del tipo flotante o entero.



## 3.6 Módulos, funciones y procedimientos de los algoritmos

- Un módulo es un algoritmo o programa que puede ser utilizado en conjunto con otros para crear algoritmos o programas más grande.
- Ejemplo: un modular es un sistema de audio que tiene un componente para tocar cintas, discos de vinil, CDs, etc. Los módulos son distinguibles.



## Módulos

- Los módulos ayudan a hacer los programas más fáciles de leer, más extensibles, ayudan a escribir menos código.
- Los módulos existen en los diferentes lenguajes de programación sin importar su paradigma de construcción: programación estructurada, programación orientada a objetos, programación orientada a eventos, etc.



## Módulos

- En programación estructurada los dos tipos más importantes de módulos son las funciones y los procedimientos.
- Las funciones realizan actividades que al final devuelven un valor, mientras que los procedimientos realizan actividades sin devolver valor



## Procedimiento

```
PROCEDIMIENTO imprimir_numero
ENTRADA:
    num1 : entero
INICIO
    IMPRIME "El número que se introdujo fue:", num1
FIN

ALGORITMO
    imprimir_numero()
```



## Función

```
FUNCION area_triangulo • Llamada desde el algoritmo:
ENTRADA:
    base, altura: real
SALIDA:
    area: real • Resultado = area_triangulo(base, altura);
INICIO
    area = base*altura/2
FIN
```



## ¿Preguntas?

