TECNOLÓGICO UNIVERSITARIO PLAYACAR

VALORES · CULTURA · VANGUARDIA EDUCATIVA

Estructuras Algorítmicas

Tadeo Manzanero Arjona *7281*

Docente: Ingeniero aeroespacial militar grado especial Daniel Conrado

Playa del Carmen, Q. Roo a

04 de noviembre de 2024

Estructuras Algorítmicas: La Sinfonía de la Lógica

Las estructuras algorítmicas son los pilares fundamentales sobre los que se construye la lógica de la programación. Son las herramientas que permiten organizar y controlar el flujo de ejecución de las instrucciones en un programa, determinando el orden en que se realizan las operaciones y cómo se procesan los datos.

A continuación, exploraremos en detalle las estructuras algorítmicas más comunes, sus características y cómo se interrelacionan para crear programas complejos y eficientes.

1. Estructura Secuencial: La Melodía Lineal

La estructura secuencial es la más básica y representa la ejecución de instrucciones en un orden lineal, una tras otra. Es como una melodía que fluye de principio a fin sin saltos ni repeticiones.

Características:

- **Simpleza:** Fácil de entender y seguir.
- **Orden estricto:** Las instrucciones se ejecutan en el orden en que aparecen.
- **Limitación:** No permite la toma de decisiones ni la repetición de acciones.

Ejemplo:

Inicio

Leer nombre

Leer edad

Mostrar "Hola " + nombre + ", tienes " + edad + " años."

Fin

2. Estructura Condicional: La Bifurcación del Camino

La estructura condicional introduce la capacidad de tomar decisiones en un programa. Permite ejecutar diferentes bloques de código según se cumpla o no una determinada condición. Es como un camino que se bifurca en dos direcciones, dependiendo de una señal.

Tipos de estructuras condicionales:

- **Simple:** Si (condición) entonces (instrucciones)
- **Doble:** Si (condición) entonces (instrucciones) sino (instrucciones)

- **Múltiple:** Si (condición1) entonces (instrucciones) sino si (condición2) entonces (instrucciones) ... sino (instrucciones)

Ejemplo:

Inicio

Leer calificación

Si (calificación >= 60) entonces

Mostrar "Aprobado"

Sino

Mostrar "Reprobado"

Fin Si

Fin

3. Estructura Iterativa (Bucles): El Ritmo Repetitivo

La estructura iterativa permite repetir un bloque de instrucciones un número determinado de veces o mientras se cumpla una condición. Es como un ritmo que se repite en una canción, creando un patrón.

Tipos de estructuras iterativas:

- **Para (For):** Se repite un número fijo de veces.
- **Mientras (While):** Se repite mientras una condición sea verdadera.
- **Hacer-Mientras (Do-While):** Se ejecuta al menos una vez y se repite mientras una condición sea verdadera.

Ejemplo:

Inicio

Para i desde 1 hasta 10 hacer

Mostrar i

Fin Para

Fin

4. Estructuras Anidadas: La Complejidad Armónica

Las estructuras algorítmicas pueden anidarse, es decir, una estructura puede contener a otra en su interior. Esto permite crear programas más complejos y con

una lógica más elaborada, como una sinfonía con múltiples melodías y ritmos entrelazados.

Ejemplo:

```
Inicio
 Leer número
 Si (número > 0) entonces
  Mientras (número > 0) hacer
   Mostrar número
   número = número - 1
  Fin Mientras
 Sino
  Mostrar "El número debe ser positivo"
 Fin Si
Fin
```

5. Modularidad: La Orquestación del Código

La modularidad consiste en dividir un programa en módulos o subprogramas más pequeños y manejables. Cada módulo realiza una tarea específica y puede ser llamado desde otras partes del programa. Esto facilita la organización, la reutilización del código y el trabajo en equipo, como una orquesta donde cada instrumento tiene su propia partitura.

Tipos de módulos:

- **Funciones:** Devuelven un valor.
- **Procedimientos:** No devuelven un valor.

Beneficios de la modularidad:

- **Claridad:** Facilita la comprensión del código.
- **Reutilización:** Los módulos pueden ser utilizados en diferentes programas.
- **Mantenimiento:** Facilita la corrección de errores y la modificación del código.
- **Trabajo en equipo:** Permite que diferentes programadores trabajen en diferentes módulos.

Representación de Algoritmos: Partituras Visuales

Los algoritmos pueden representarse de forma visual para facilitar su comprensión y análisis. Las dos formas más comunes son:

- **Diagramas de flujo:** Utilizan símbolos gráficos para representar las diferentes acciones y el flujo de ejecución.
- **Pseudocódigo:** Utiliza un lenguaje similar al lenguaje natural para describir las instrucciones del algoritmo.