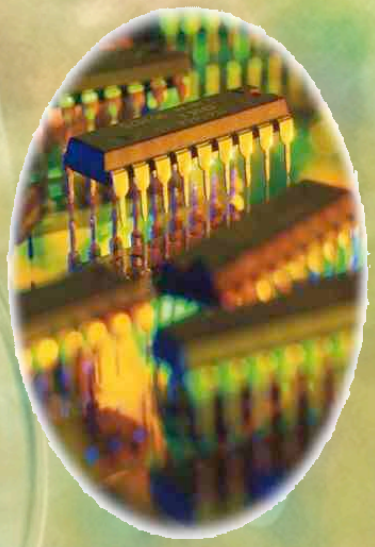
TEMA

1

Introducción a la Programación



Dept. Ciencias de la Computación e I.A.

Universidad de Granada

Índice

- Resolución de Problemas con Ordenadores
- Algoritmo
- Metodología de la programación
- Lenguajes de programación
- Entornos integrados de desarrollo

Conceptos Básicos

- La solución a un problema debe establecerse en términos de una secuencia de pasos computacionales (programa)
- Su especificación se hace a través de notaciones sistemáticas conocidas como lenguajes de programación
- La programación es la tarea de realizar programas para resolver problemas mediante ordenador
- Existen metodologías que permiten que la programación sea lo más eficaz en cuanto al desarrollo y mantenimiento

Los ordenadores son "tontos"

- Ejecutan rapida y precisamente operaciones lógicas y matemáticas
- Cumplen obedientemente lo que les decimos (no siempre igual a lo que queremos!)
- Errores pequeños pueden provocar fallos graves

¿Qué es un Algoritmo?

Definición: *método para resolver un problema mediante una serie de pasos:*

1. *Precisos*: (indicar el orden de ejecución en cada paso).
2. *Definidos* (si el algoritmo se prueba dos veces, en estas dos pruebas, se debe obtener el mismo resultado).
3. *Finitos* (el algoritmo tiene que tener un número determinado de pasos).

Los algoritmos se pueden expresar por fórmulas, diagramas de flujo, y pseudo códigos. Está última representación es la mas utilizada por su sencillez y parecido al lenguaje humano.

Un ejemplo de Algoritmo: ver una película

- Buscar el dvd de la película
- SI el televisor y el dvd se encuentran apagados, encenderlos
- Sacar el dvd de su caja
- Introducirlo en el reproductor
- Tomar el mando del televisor y el dvd (o el mando universal!)
- Ir al sofá
- Ponerme cómodo
- Darle al "play"

Metodología de la Programación

La utilización de conceptos y técnicas adecuadas de programación, es fundamental para obtener programas que cumplan los siguientes requisitos:

- Legibles: comprensibles por cualquier programador
- Modificables: su estructura debe permitir modificaciones
- Depurables: debe ser fácil la localización y corrección de errores

Y como elementos adicionales, se obtiene:

- Reducción de los costos de mantenimiento.
- Aumento de la productividad del programador.
- Los programas quedan mejor documentados internamente.

Metodología de la Programación

Si no seguimos un método de programación riguroso:

- Los programas son rígidos y difíciles de adaptar a nuevos requerimientos
- Los programadores gastan mucho tiempo corrigiendo sus errores
- La comunicación entre programadores es muy difícil
- Los programas y módulos son poco reusables
- Existen deficiencias en la documentación

Resolución de problemas con el Ordenador

El "ciclo de vida" del software

Análisis de Requerimientos

Esta fase consta de dos partes orientadas a responder QUE es lo que hay que hacer:

1. Definición del Problema: identificar claramente que es lo que se quiere resolver
2. Análisis del problema: se basa en identificar
   - La fuente y los tipo de datos para las entradas
   - Interacción de los datos y su transformación
   - Destino y formato de salida

El análisis se realiza empleando el lenguaje "natural"

Etapa de Diseño

En esta fase resolvemos el COMO se hace lo especificado en el análisis de requerimientos:

- "ideamos" un algoritmo
- utilizamos "pseudocódigo" para su descripción
- Aplicamos la técnica *divide y vencerás*: la resolución de un problema complejo se realiza diviendo el problema en subproblemas
- Aplicamos diseño descendente o modular
- *El diseño del algoritmo es independiente del lenguaje de programación en el que se vaya a codificar posteriormente*

Implementación y Validación

Implementación: se pasa del diseño obtenido en la etapa anterior, a una descripción escrita en un lenguaje de programación (un "dialecto" que pueda comprender el ordenador). Esta descripción (programa) debe "compilarse" y "ejecutarse" para pasar a la fase de validación.

Validación: es el proceso de identificar y eliminar errores para poder responder a dos preguntas:

- Construimos el software correcto ? (contra el usuario)
- Lo construimos correctamente? (contra la especificación)

Lenguajes de Programación (1)

Un lenguaje de programación es una herramienta para producir software. Se compone de un conjunto de reglas, símbolos y signos que permiten la comunicación con el ordenador.

Existen 3 categorías:

1) Lenguajes de máquina: "entendible" por el ordenador.

- Cadenas de ceros y unos
- Alta velocidad de ejecución
- Muy complejo de utilizar
- Los programas resultantes son difíciles de leer
- Cada procesador tiene un lenguaje máquina propio. No son trasportables

Lenguajes de Programación (2)

2) Lenguajes ensambladores:

- Utilizan códigos mnemotécnicos para representar cadenas de 0's y 1's
- Aumentan la comprensibilidad de los programas
- Son dependientes del procesador
- Cada instrucción, representa una operación del ordenador

Surgen los "*Ensambladores*": software que traduce los programas en ensamblador a programas en lenguaje de máquina para su ejecución

Lenguajes de Programación (3)

3) Lenguajes de alto nivel

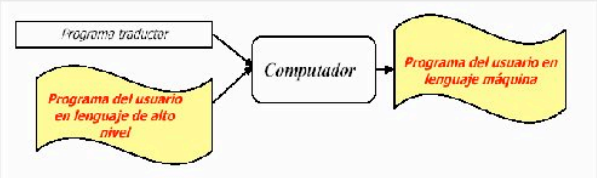
- Proporcionan un repertorio de instrucciones amplio, potente, y fácilmente asimilable
- Los programas son mucho más legibles y claros
- Son fáciles de aprender y mantener
- Son transportables
- FORTRAN, COBOL, C, C++, PASCAL, etc.

```
'BASIC PROGRAM
'AVERAGING INTEGERS ENTERED THROUGH THE KEYBOARD
CLS
PRINT 'THIS PROGRAM WILL FIND THE AVERAGE OF INTEGERS YOU ENTER'
PRINT 'THROUGH THE KEYBOARD. TYPE 999 TO INDICATE END OF DATA."
PRINT
SUM=0
COUNTER=0
PRINT 'PLEASE ENTER A NUMBER"
INPUT NUMBER
DO WHILE NUMBER <> 999
    SUM=SUM+NUMBER
    COUNTER=COUNTER+1
    PRINT 'PLEASE ENTER THE NEXT NUMBER"
    INPUT NUMBER
LOOP
AVERAGE=SUM/COUNTER
PRINT 'THE AVERAGE OF THE NUMBERS IS' AVERAGE
END
```

```
// C++ PROGRAM
// AVERAGING INTEGERS ENTERED THROUGH THE KEYBOARD

#include <iostream.h>
main ()
{
  float average;
  int number, counter = 0, int sum = 0;
  cout << "THIS PROGRAM WILL FIND THE AVERAGE OF INTEGERS YOU ENTER\ n";
  cout << "THROUGH THE KEYBOARD.  TYPE 999 TO INDICATE END OF DATA. \ n";
  cout << "PLEASE ENTER A NUMBER";
  cin >> number;
  while (number !=999)
    {
      sum := sum + number
      ocunter ++;
      ocut << "\nPLEASE ENTER THE NEXT NUMBER"
      cin >> numbe;
    }
  average = sum / counter
  cout << "\nTHE AVERAGE OF THE NUMBERS IS " << average
}
```

1. Tanto los lenguajes de alto nivel como los de bajo nivel, no son entendibles directamente por la máquina.
2. Necesitan ser traducidos a instrucciones en lenguaje de máquina
3. Es necesario disponer de un "traductor" o interfase con el lenguaje de máquina para que el programa sea ejecutable.

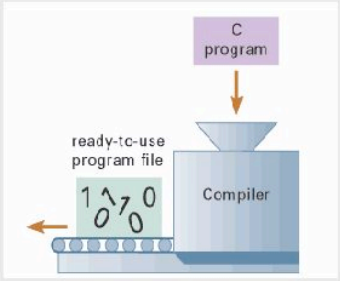


Existen dos tipos fundamentales de traductores:

- Compiladores
- Interpretes

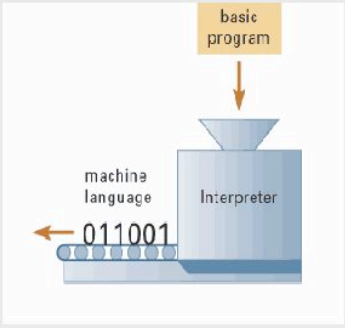
Compiladores

Software que traduce el programa inicial (programa fuente) escrito en un lenguaje de alto nivel, a un programa (programa objeto) en lenguaje de máquina factible de ser ejecutado



Intérpretes

Analizan, traducen y ejecutan una a una las instrucciones del programa fuente; no se analiza una instrucción hasta que la anterior se haya ejecutado. Los intérpretes no generan programa objeto.



Entornos Integrados de Desarrollo

Un "IDE" es un software que incorpora todas las herramientas necesarias para el desarrollo de nuestros programas

Por herramientas entendemos:

- Editor
- Compilador
- "Linker"
- Depuración
- Ejecución

En prácticas usaremos el Dev-C++