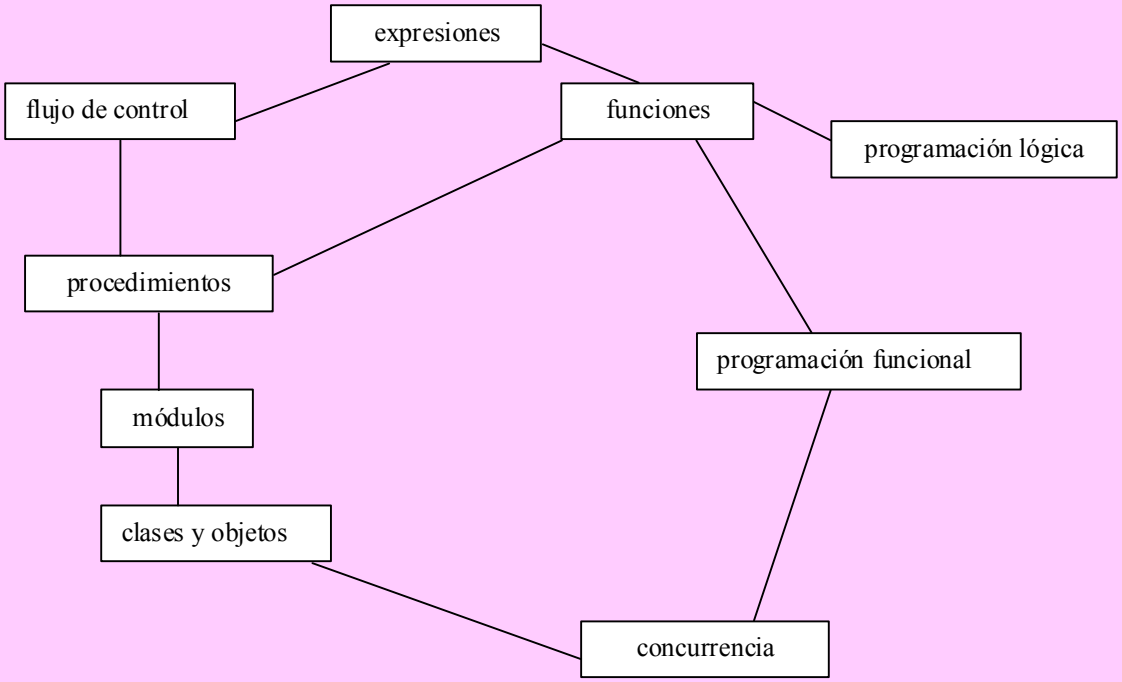
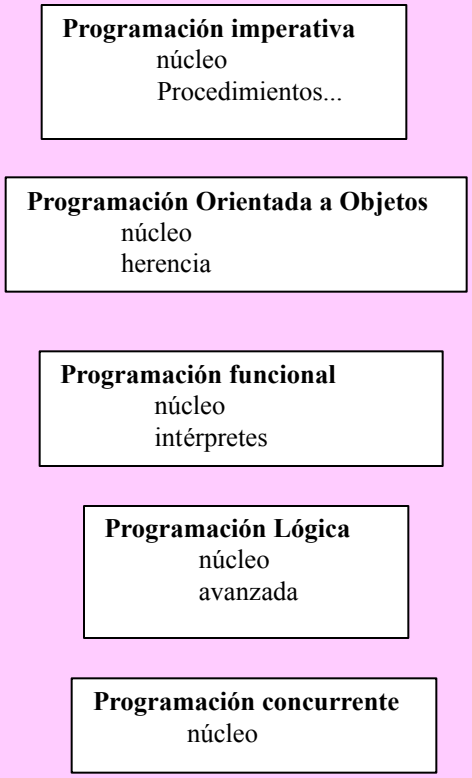
Lenguajes de programación

Ana Lilia Laureano Cruces


expresiones
flujo de control
funciones
programación lógica
procedimientos
programación funcional
módulos
clases y objetos
concurrencia


Programación imperativa
núcleo
Procedimientos...
Programación Orientada a Objetos
núcleo
herencia
Programación funcional
núcleo
intérpretes
Programación Lógica
núcleo
avanzada
Programación concurrente
núcleo

Definiciones

- Un programa es la especificación de una tarea de computación.
- Un lenguaje de programación es una notación para escribir programas.
- El desarrollo de los lenguajes se ha basado en el conocimiento de que la complejidad puede manejarse imponiendo alguna estructura a los datos, a las operaciones, a los programas e incluso a las descripciones de un lenguaje.

La Función de la Estructura en la Programación

- Dijkstra [1972] Es poco probable el establecimiento de la corrección de un programa mediante pruebas, a menos que se tome en cuenta su estructura interna.
- La única esperanza es diseñar cuidadosamente un programa de manera que su corrección pueda entenderse en términos de su estructura.

- El arte de la programación es el arte de organizar la complejidad y

- Debemos organizar los cálculos de manera que nuestros limitados sentidos sean suficientes para garantizar que el cómputo arroje los resultados esperados.

Qué es la programación

- En computación se desarrolla más trabajo de programación que de desarrollo (creación inicial a la verificación de un programa)

- El lenguaje debe ayudarnos a escribir buenos programas; un programa es bueno si es fácil de leer, fácil de entender y fácil de modificar.

- Técnicas para el manejo de programas pequeños no son crecientes (en general).

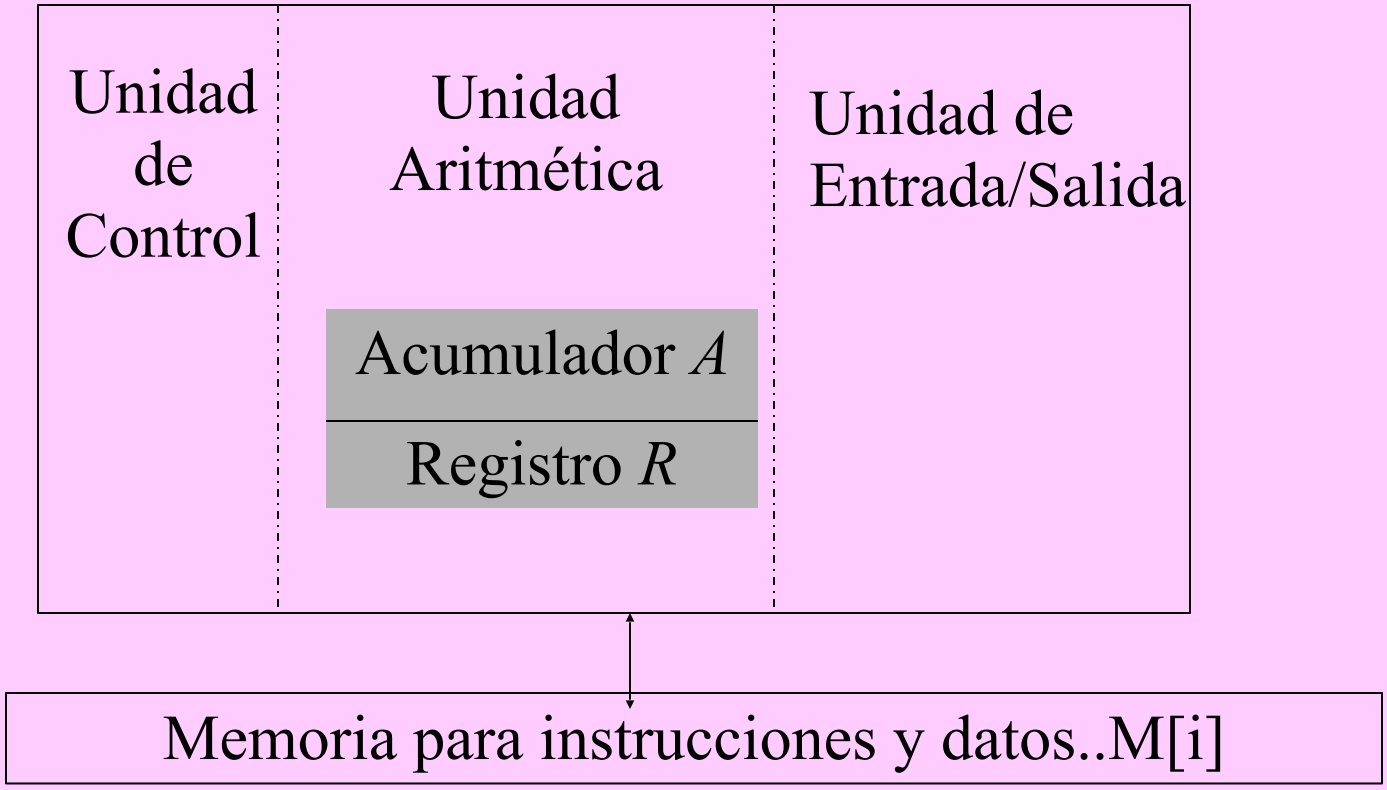
- Las dificultades comienzan cuando el efecto del cambio puede extenderse a través de un programa muy grande, introduciendo errores en un lugar olvidado. Pueden permanecer sin detección durante años.

- La estructura y la organización son la clave para manejar programas muy grandes.
- La legibilidad de un programa puede mejorarse organizandolo de forma que cada parte pueda entenderse en forma relativamente independiente del resto.
- La estructura nos ayuda a mantener la situación dentro del límite de la atención humana.

- Miller [1967] observó que la gente es capaz de recordar aproximadamente 7 cosas:
  - 7 bits (dígitos binarios)
  - palabras, colores,tonos, sabores
  - 7 maravillas, 7 pecados capitales, 7 edades del hombre
  - la memoria esta limitada por el número de símbolos o unidades que pueda manejar y no por la información que representen esos símbolos

La máquina de Von Neumann

- Los orígenes de los lenguajes de programación se encuentran en la máquinas
- se diseño a finales de los 40's en el Instituto de Estudios Avanzados de Princeton
- Las actuales tienen mucho en común = arquitectura Von Neumann
- Lenguaje de máquina = código (texto en un programa)


Unidad de Control
Unidad Aritmética
Unidad de Entrada/Salida
Acumulador A
Registro R
Memoria para instrucciones y datos..M[i]

Las características de la máquina

- **Datos:** enteros eran la única forma.
- **Operaciones aritméticas:** *sumar, restar, multiplicar, dividir* y *tomar el valor absoluto de un número*. El resultado de *suma o resta* se colocaba en un registro llamado *acumuldor*.
- **Asignaciones a localidades de memoria:** a una localidad se le podía asignar el valor contenido en el *acumulador* y reemplazaba el valor anterior.

- **Flujo de control:** el flujo normal del control de una instrucción a la sigueinte podía ser interrumpido por una instrucción de salto incondicional (*go to*).
- **Justificación del *go to*:** la utilidad de un computador automático reside en la posibilidad de usar en forma repetida una secuencia de instrucciones, en donde el número de iteraciones *puede asignarse previamente o depender de los resultados de un cálculo.*

- Código de máquina es ininteligible:
  - 00000010101111001010
  - 00000010111111001000
  - 00000011001110101000
  - LOAD I
  - ADD J
  - STORE K
  - k = i + j

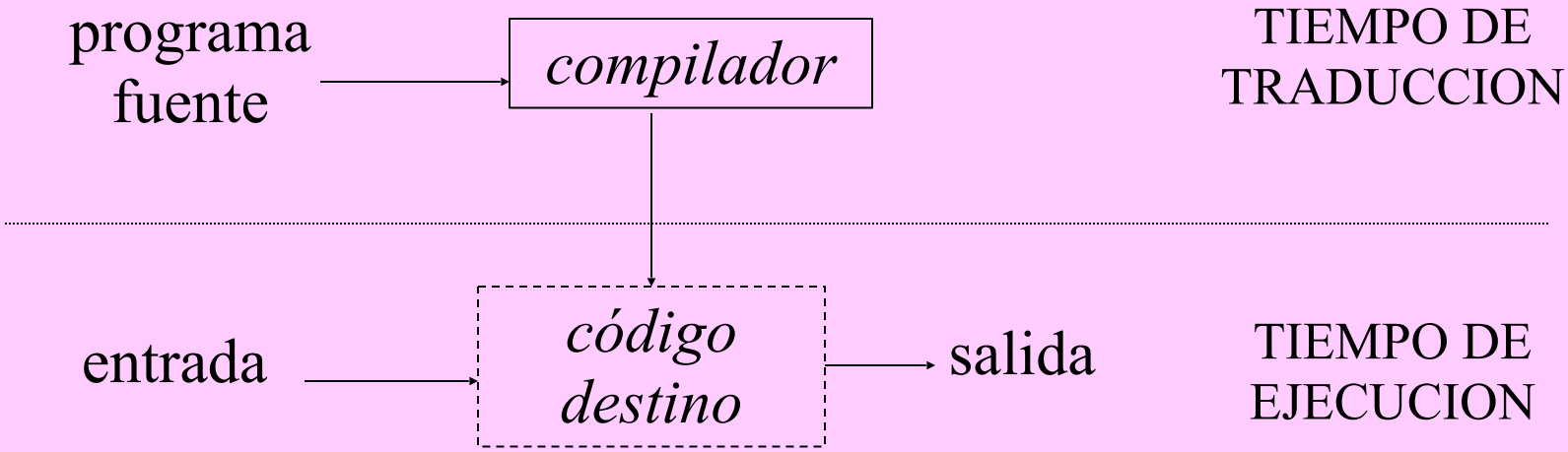
- El lenguaje ensamblador: es una variante del lenguaje de máquina. Se manejan identificadores en vez de códigos reales (valores, localidades de almacenamiento y operaciones)

- El lenguaje de máquina y los ensambladores se les conoce como *lenguajes de bajo nivel*

- El alto costo de creación de código ensamblador o máquina Yel desarrollo de FORTRAN = FORmula TRANslation.
- Backus [1975], tenía como meta:
  - Permitir al programador especificar un procedimiento numérico mediante el uso de un lenguaje conciso como el de las matemáticas.

- A partir de esta especificación:
  - un programa en lenguaje de máquina
  - depuración (localización y corrección de errores) se reduce
  - se esperaba reducir a un quinto el desarrollo de programas.

- El traductor de un lenguaje de máquina o ensamblador se conoce como *compilador*.



Problemas

- *Tiempo de traducción*: se necesita tiempo de máquina para compilar un programa fuente y convertirlo en código destino.

- *Tiempo de ejecución y necesidad de espacio mayores*: el código destino creado por el compilador suele ejecutarse con mayor lentitud y ocupa más espacio que un código escrito directamente.

Preocupación

- Backus pensó que la aceptación de FORTRAN se hallaría en serios problemas al tardar el doble de tiempo en la ejecución que su contraparte escrita en código de máquina.
  - Notación fácil de leer
  - la escritura tenía una notación cercana a la que describía originalmente el problema
  - eran portátiles (intercambio y creación de acervos)

- Con respecto al lenguaje C:
  - Ritchie concluye: aunque un estudio profundo de la mayor cantidad de espacio y tiempo utilizados debido al uso de C, pudiera resultar interesante, sería irrelevante ya que sin importar sus resultados, no provocaría que volviéramos a escribir en lenguaje ensamblador.

Revisión de la eficiencia

- La eficiencia de un programa depende de las decisiones tomadas en todos los niveles:
  - concepción
  - elección de estructuras de datos
  - algoritmos
- La legibilidad y la capacidad de modificar los programas contribuyen a la eficiencia.

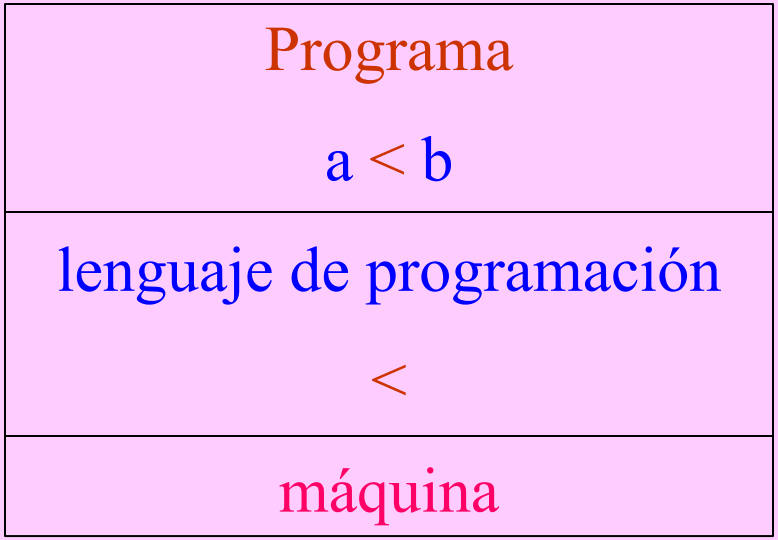
- El desarrollo de un programa eficiente puede considerarse como una secuencia de cambios evolutivos guiados por la realización de un análisis.
- El refinamiento de código se realiza con la mejora de puntos críticos, que son las pequeñas partes muy utilizadas en donde un programa emplea la mayor parte de su tiempo de ejecución.

- Detector de perfiles (variante de compiladores): detecta los *puntos críticos*.

- La codificación cuidadosa de los *puntos críticos* mejora significativamente el tiempo de ejecución.

- En general un lenguaje de programación es una extensión de la máquina en que se apoya; y un programa es una extensión del lenguaje de programación.
- Un lenguaje reconstruye la máquina para proporcionar más recursos, y un programa reconstruye el lenguaje para proporcionar recursos más cercanos al problema que debe resolverse.

Cada capa amplía las posibilidades de la capa inferior



Recursos de un lenguaje

- *Modelo de computación* : capacidades de la máquina que lo sustenta o bien proporcionar una aproximación diferente (lenguajes concurrentes)

- *Tipos de datos y operaciones* : constructores (registros, arreglos) del lenguaje que permiten estructurar los valores básicos que posee la máquina como son: caracteres, enteros y números reales. Cada tipo estructurado tiene asociado un conjunto de operaciones, que permite manipular sus componentes.

- *Recursos de abstracción* :
  - *funciones:* son abstracciones de las operaciones integradas (sumas, restas, multiplicaciones), *rcua.*
  - *procedimientos:* son abstracciones de las acciones integradas (asignación), *ordena*.
  - Definir TAD's

- *Verificación y validación* :
  - la *verificación en tiempo de ejecución*; detecta errores antes de que se ejecuten
  - la *verificación* permite la validación del encapsulamiento.

La estructura de un programa

- Existe más de una forma para solucionar un programa de aquí que exista más de una forma para estructurar un programa.

- Una forma de estructurar es a través del refinamiento de sucesivo: descomponer problemas en subproblemas más simples que se puedan resolver de una manera relativamente independiente.

- El diseño de una descomposición apropiada es la parte más difícil de ésta técnica; debido a que la descomposición no es obvia en un principio.

- Entran en juego el paso de parámetros

Estructura sintáctica

- La sintaxis de un lenguaje especifica cómo están constituidos los programas en dicho lenguaje.

- La estructura sintáctica es la estructura impuesta por la sintaxis del lenguaje. Constituye la herramienta primaria para trabajar con el lenguaje.

- La sintaxis de un lenguaje de programación se especifica usando alguna variante de la notación conocida como gramática independiente del contexto.

- BNF (Backus-Naur Form) o BNFE (extendida)

- <número real> ::= <secuencia-dígitos>. <secuencia-dígitos>
- <secuencia-dígitos> ::= <dígito> | <dígito> <secuencia-dígitos>
- <dígito> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

- < > ; *constructores*
- ::= ; *es*
- | ; *o*