escuelaingenieríaindustrial
PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAISO

Algoritmos

Prof. Franco Guidi Polanco
Escuela de Ingeniería Industrial
Pontificia Universidad Católica de Valparaíso, Chile
fguidi@ucv.cl

Contenido

- Definiciones básicas y componentes de un algoritmo
- Representación de algoritmos mediante diagramas de Nassi-Shneiderman (N-S)
- Representación de algoritmos mediante seudocódigo

escuelaingenieríaindustrial
PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAISO

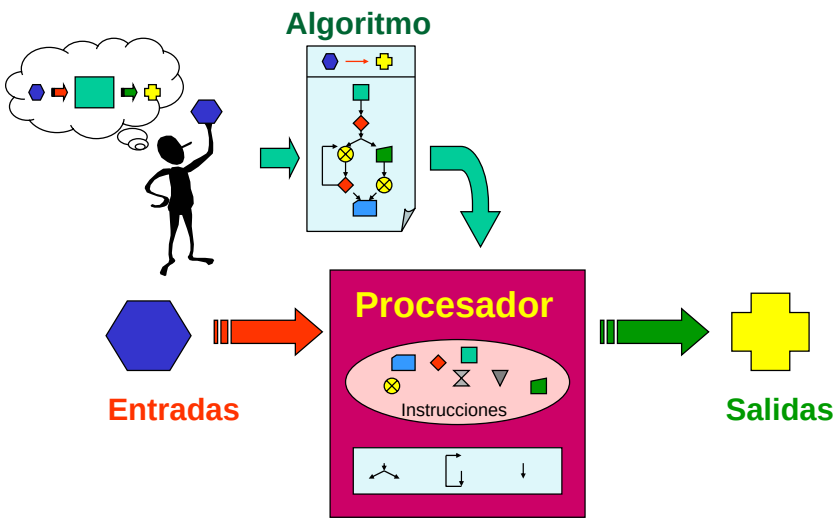
Definiciones básicas y componentes de un algoritmo

Prof. Franco Guidi Polanco
Escuela de Ingeniería Industrial
Pontificia Universidad Católica de Valparaíso, Chile
fguidi@ucv.cl

Definiciones

- **Algoritmo:** es una secuencia ordenada de pasos, carentes de ambigüedades, que conducen a la solución de un problema dado (de Al-Khôwarizmi, matemático y astrónomo árabe del s. IX).
- Todo algoritmo debe ser:
  - **Preciso:** pasos no ambiguos y en secuencia específica.
  - **Definido:** a partir de las mismas entradas debe generar siempre los mismos resultados.
  - **Finito:** su ejecución debe completarse en un número finito de pasos.

El algoritmo es un modelo



Características de un procesador estándar

- Conoce un conjunto limitado de instrucciones elementales.
- Dispone de un conjunto reglas para ordenar las instrucciones elementales (estructuras de control).
- Acepta entradas y es capaz de generar salidas.
- Tiene memoria.

Solución de problemas por computador

- La solución de un problema mediante el uso de un computador, tiene las siguientes etapas:
  - Análisis del problema
  - Desarrollo de un algoritmo:
    - Diseño del algoritmo
    - Verificación del algoritmo
  - Desarrollo de un programa:
    - Codificación del algoritmo (programación)
    - Pruebas
  - Utilización del programa

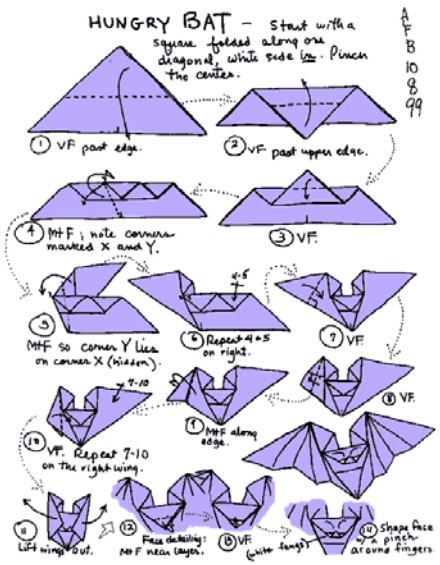
Ordenamiento de las instrucciones

- Las instrucciones de un algoritmo se pueden organizar mediante tres estructuras de control:

Secuencia | Decisión | Iteración

Estructuras de control: Secuencia

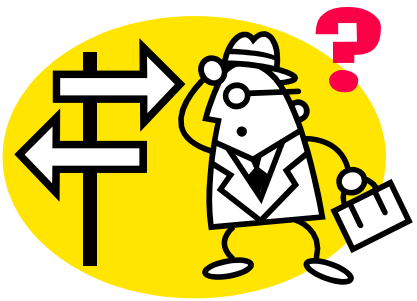
Primero hacer A,
luego hacer B,
luego hacer C,
…
y finalmente hacer Z.



Origami.
Secuencia de pliegues
para hacer un murciélago

Estructuras de control: Decisión

Si se cumple la condición **<*cond*>**,
entonces hacer algo,
en caso contrario hacer otra cosa.



Estructuras de control: Iteración

Mientras se cumpla la condición **<*cond*>**,
repetir hacer algo.

o

Repetir hacer algo,
mientras se cumpla la condición **<*cond*>**.

Mientras falten copias por sacar,
imprimir una copia del documento.

escuelaingenieríaindustrial
PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAISO

Representación de algoritmos mediante diagramas de Nassi-Shneiderman

Prof. Franco Guidi Polanco
Escuela de Ingeniería Industrial
Pontificia Universidad Católica de Valparaíso, Chile
fguidi@ucv.cl

Presentación

- Diagramas creados para apoyar la didáctica de la programación estructurada.
- Impiden la especificación de “saltos” en el flujo de control del algoritmo (asociados a la instrucción GOTO, y habilitados por diagramas de flujo tradicionales)

Origen

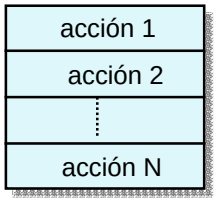
- Desarrollados y publicados por Ben Shneiderman y Ike Nassi
- Concebidos por Shneiderman en 1972 mientras participaba en una conferencia de la ACM sobre programación.

Dr. Ike Nassi

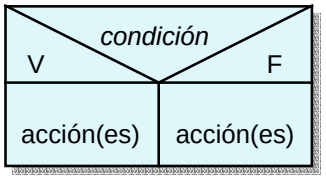
Dr. Ben Shneiderman

Representación de estructuras de control

- Secuencia

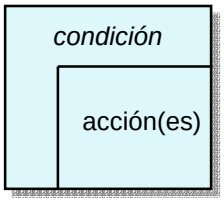


- Decisión

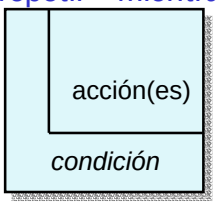


- Iteración

mientras -repetir



repetir - mientras

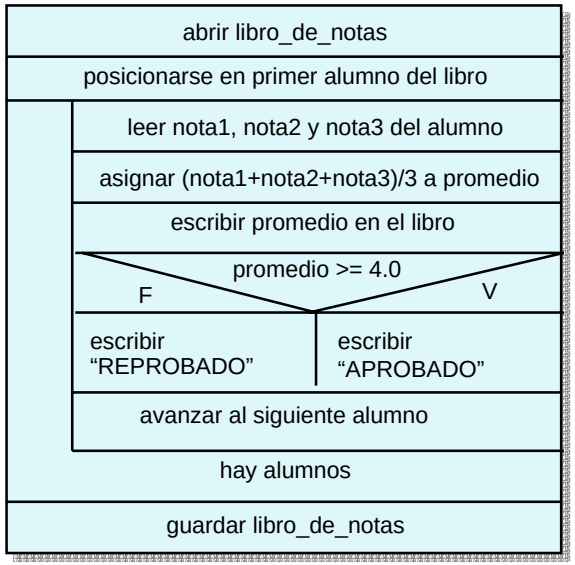


Otras estructuras de control

- Los diagramas de Nassi-Shneiderman permiten la especificación de otras estructuras de control.
- Tales estructuras no serán estudiadas en este curso.

Ejemplo

- Ejemplo: calcular promedios y determinar condición final de alumnos de un curso.



Un poco de historia: publicación de los Diagramas de Nassi-Shneiderman

- Nassi y Shneiderman propusieron su trabajo denominado diagramas de flujo estructurados (*structured flowcharts*) para ser publicado en la revista Communications of the ACM.
- La respuesta del editor en base a un revisor no se hizo esperar:

My first thought was to write a referees report which would by its sarcastic nature be funny. For example, I thought of writing that it was a sound, useful theory, but it wasn't practical because it would be difficult to design flowcharting templates to be manufactured and sold.

I guess, however, that it is best to come right out and say that I feel the best thing the authors could do is collect all copies of this technical report and burn them, before anybody reads them. My opinion is that it shows the inexperience and ignorance of the authors with respect to programming in general, and their misunderstanding of so-called structured programming.

Un poco de historia: publicación de los Diagramas de Nassi-Shneiderman (cont.)

- Nassi y Shneiderman decidieron reenviar su artículo a la ACM, ahora a ACM SIGPLAN Notices, donde fue publicado en 1973, bajo el título "***Flowchart Techniques for Structured Programming***" (SIGPLAN Notices 8, 8 August, 1973).
- Desde entonces se han desarrollado varias extensiones, herramientas y aplicaciones
- Los diagramas de Nassi-Shneiderman tuvieron bastante acogida en Alemania, donde se les conoce como Struktogramme, y estandarizados en 1985 mediante la norma DIN 66261.

El "robo" de los diagramas de Nassi-Shneiderman

- Nassi y Shneiderman compartieron sus ideas con algunos colegas, entre ellos Ned Chapin.
- Chapin presentó tales diagramas (o derivaciones de ellos) como un trabajo propio, bajo el nombre de "***Chapin Charts***". Shneiderman, e su sitio web denuncia este hecho como un "robo de la idea"[1].
- Hoy, sin embargo, son reconocidos mundialmente bajo el nombre de Nassi-Schneiderman Diagrams (o simplemente como NS Diagrams), a pesar de que ellos los habían bautizado originalmente como "structured flowcharts")

Dr. Ned Chapin

[1] Shneiderman B., **Rejection Letter from the Communications of the ACM**, June 12, 2003. Disponible via web: http://www.cs.umd.edu/hcil/members/bshneiderman/nsd/rejection_letter.html

escuelaingenieríaindustrial
PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAISO

Desarrollo de algoritmos: nuestros acuerdos sobre un procesador estándar

Prof. Franco Guidi Polanco
Escuela de Ingeniería Industrial
Pontificia Universidad Católica de Valparaíso, Chile
fguidi@ucv.cl

Problema

- Se pide desarrollar un algoritmo para marcar un número telefónico.
- Tres propuestas:

(a)	(b)	(c)
Levantar auricular	Levantar auricular	Llamar al número indicado
¿Hay números por discar?	Discar números	
Discar número		

¿Cuál algoritmo está correcto?

La necesidad de definir un procesador "estándar"

- Por fines didácticos necesitamos definir un procesador, de tal forma de hacer nuestras propuestas comparables.
- Para efectos de este curso nuestro procesador será suficientemente cercano a lo que ofrecen los lenguajes de programación:
- Definiciones corresponden a:
  - Tipos de datos
  - Operadores
  - Variables
  - Operadores
  - Funciones
  - Procedimientos
  - Expresiones
  - Estructuras de control

Componentes de un algoritmo

- Tipos de datos:
  - Numéricos:
    - Ej. **3.0**, **1.5**, **-5.0**, **4E-3**, etc.
  - Caracter / cadena de caracteres: Ej. **"A"**, **"Hola"**, **"10"**, **"01/09/97"**, etc.
- Operadores:
  - Aritméticos: **+**, **-**, *****, **/**, **%** (módulo).
  - Lógicos: **&&** (AND), **||** (OR), **!** (NOT).
  - De relación: **==**, **>**, **<**, **<>**, **>=**, **<=**.
  - De asignación: **=**

Componentes de un algoritmo

- Variables: almacenan valores de alguno de los tipos de datos definidos (numérico o cadena de caracteres). Se identifican mediante un nombre que "debe" comenzar con un caracter alfabético, pudiendo ser los restantes letras, dígitos o el guión bajo "_" (*underscore*).
  - **Correctos**: edad, DiaSemana, NOMBRE, Factor01, Hra_Tarde
  - **Incorrectos**: 2oNombre, min+seg, km/hra, Nota#1, Hra's, dia(s)
- Los valores pueden ser modificados durante la ejecución del algoritmo.

Componentes de un algoritmo

- Funciones: funciones matemáticas que reciben uno o más argumentos y retornan un resultado. Ej.: **sin**(x), **cos**(x), **ln**(x), **exp**(x).
- Procedimientos: son acciones que pueden ser invocadas en el algoritmo. Los procedimientos estándares permiten el ingreso de datos (**Read:**) y su presentación (**Write:**). Ej.:
  - **Read:** nombre
  - **Write:** nombre
  - **Write:** 7

Componentes de un algoritmo

- Expresiones: combinan variables y funciones mediante operadores. Pueden ser:
  - **Aritméticas**: al ser evaluadas retornan un valor numérico.

(5 + edad) * 2	si **edad** es una variable numérica
(base * altura) / 2	si **base** y **altura** son variables numéricas
sin(angulo) * 1.5	si **angulo** es variable numérica

  - **Lógicas**: al ser evaluadas retornan un valor lógico.

edad > 20	si **edad** es una variable numérica
(ancho >=10) y (alto >= 5)	si **ancho** y **alto** son var. numéricas
(nombre == "Pedro")	si **nombre** es var. cadena de caracteres

Componentes de un algoritmo

- Estructuras de control:
  - Secuencia:
    acción 1
    acción 2
    ....
    acción N
  - Decisión:
    si ***condición*** **entonces** *acciones* **si no** *otras acciones*.
  - Repetición:
    mientras ***condición*** **repetir** *acciones*
    repetir *acciones* **mientras** ***condición***

Importante: ***condición*** representa una expresión lógica.

Relaciones entre los componentes de un algoritmo