



Algoritmos

Introducción a la Programación
Facultad de Informática
Univ.Nac. del Comahue



Unidad 2

- **Algoritmos**
- Resolución de Problemas por Computadora



Algoritmos⁽¹⁾

Un **algoritmo es un modelo** de la resolución de un problema, o mejor aún, **una clase de problemas**.

Un algoritmo es una **secuencia de operaciones precisas** (i.e. no ambiguas) para resolver un problema en un **número finito de pasos**. (Informalmente, podemos decir que es similar a una receta)


(1) La palabra algoritmo deriva del nombre del matemático Mohammed ibnMusa alKhwarizmi (780-850).




Algoritmos

Todos trabajamos continuamente con algoritmos, sólo que no somos tan formales como para llamarlos así.

Ejemplos:

Preparar Mate		1 – Agregar yerba al mate
		2 – tapar el mate y sacudir
		3 – Agregar Agua
		4 - colocar la bombilla
		5- Tomar el mate

té		1 - colocar saquito de té en una taza
		2 - llenar la taza con agua hirviendo
		3 - esperar 5 minutos
		4 - beber el té



Puntos de vista de un algoritmo

Especificación vs Ejecución

1- Especificación del algoritmo. La especificación comprende precisamente al **conjunto de acciones** que permiten resolver **una clase de problemas**. La especificación se puede pensar como una entidad **estática**.



2- Ejecución del algoritmo. Al llevar a cabo las directivas especificadas en el algoritmo, hablamos de ejecución. Durante la ejecución estamos hallando la solución de **un problema específico**. La ejecución es un **proceso dinámico**.





Especificación vs Ejecución

Clase de Problema: Calcular edad de una persona

Especificación:

$\text{edad} = \text{añoActual} - \text{añoNacimiento}$

Problemas particulares:

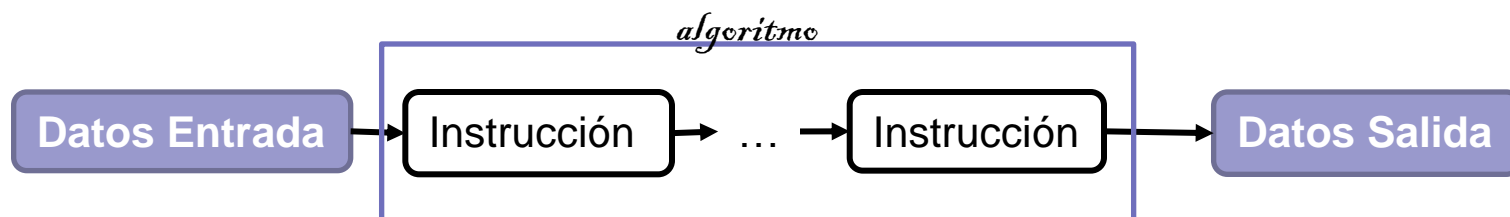
Ejecución: Edad Mengano
 $\text{edad} = 2020 - 2000$
 $\text{edad} = 20$

Ejecución: Edad Fulano
 $\text{edad} = 2020 - 1980$
 $\text{edad} = 40$



Propiedades de los Algoritmos

1) **Entrada / Salida**: Recibe Entrada y genera Salida.



2) **Precisión**: Cada paso esta precisamente explicitado. Sin ambigüedades, la orden es clara.

3) **Determinismo**: Los resultados intermedios de cada paso de ejecución son únicos y determinados sólo por las entradas y resultados del paso anterior. Podemos resumirlo como: para los mismos valores de entrada, la ejecución del algoritmo debe producir la misma salida.



Propiedades de los Algoritmos

- 4) **Correctitud**: Un algoritmo es correcto, si cumple con los tres puntos siguientes:
- * Resuelve el problema para el que se diseñó: efectividad.
 - * **Para cada entrada**, el algoritmo produce la salida deseada.
 - * El algoritmo termina en un tiempo finito.
- 5) **Generalización**: Un algoritmo es aplicable a un conjunto de entradas. (Es por esta propiedad que resuelve una clase de problemas y no un problema particular).



Unidad 2

- Algoritmos
- **Resolver Problemas por Computadora**



Resolver Problemas en Computadora

Para que una computadora pueda ser utilizada para hallar la solución a un problema es necesario que se le brinde un método para resolver dicho problema.

La tarea de programar implica la obtención de un modelo computacional para la resolución del problema.

Es decir, **lograr un conjunto de instrucciones comprensibles para la computadora** que permitan alcanzar la solución deseada para un conjunto de problemas.

Los algoritmos son el modelo computacional

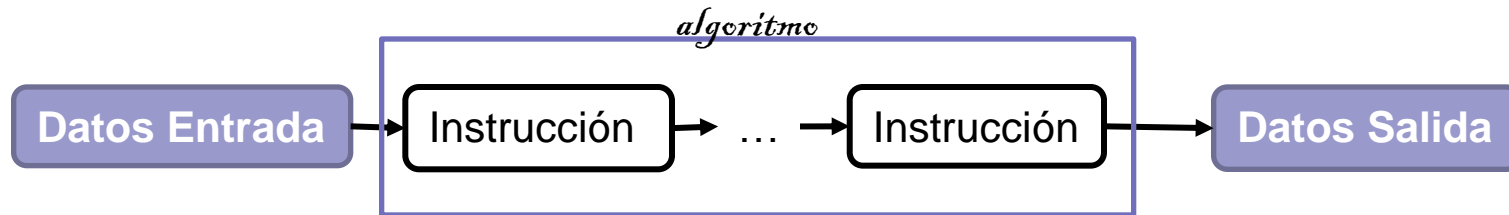


Resolver Problemas en Computadora

- Cuando el **ALGORITMO** puede ser transformado en una secuencia de instrucciones ejecutables por una computadora, **se transforma en un PROGRAMA**.
- Un **programa estará escrito en un lenguaje de programación**, es decir, en una notación formal y estricta, que podrá ser interpretada por la computadora.
- Cuando se usa una computadora es deseable minimizar el consumo de recursos, principalmente el costo en tiempo y espacio. En Cs. de la Computación, **el desafío** no es cómo resolver un problema sino cómo **resolverlo de modo eficiente**.



Resolver Problemas en Computadora



Los **datos de entrada** de un algoritmo, pueden considerarse como la descripción del estado inicial de un problema.

Los **datos de salida** de un algoritmo, pueden considerarse como la descripción del estado final de un problema.

Un **dato** es un valor. En la ejecución del algoritmo cada dato de entrada y salida corresponderá a un valor dentro de un conjunto. Dicho conjunto corresponde a un tipo de dato.



Resolver Problemas en Computadora



Un **dato** es un valor. En la ejecución del algoritmo cada dato de entrada y salida corresponderá a un valor dentro de un conjunto. Dicho conjunto corresponde a un tipo de dato.

Es probable que dependiendo del momento del algoritmo hablemos de dato o de valor:



1- Especificación del algoritmo → dato



2- Ejecución del algoritmo. → valor



Especificación vs Ejecución

Clase de Problema: Calcular edad de una persona

Especificación:

$\text{edad} = \text{añoActual} - \text{añoNacimiento}$

*Datos
de tipo Entero*

Problemas particulares:

Ejecución: Edad Mengano

$\text{edad} = 2020 - 2000$

$\text{edad} = 20$

Valores



Componentes más significativos de un programa

- **Variable**: Nombre o etiqueta usado para identificar una ubicación de memoria en donde puede almacenarse un dato. (es un contenedor que almacena un dato)
- **Tipo de Dato**: Determina el valor que puede almacenarse en una variable y las operaciones que pueden realizarse con ella. Ejemplos de tipo de datos son: **entero**, **float (real o decimal)**, **booleano (o lógico)**, **string (o cadena de caracteres)**. Otros tipo de datos más complejos son los arreglos (o arrays).
- **Instrucción**: orden o acción clara y precisa para ser ejecutada por una computadora.

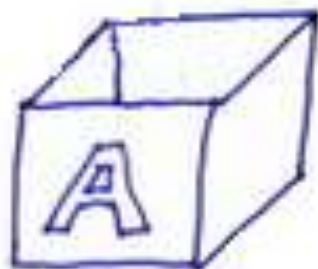


Variable

Nombre o etiqueta usado para identificar una ubicación de memoria en donde puede almacenarse un dato.

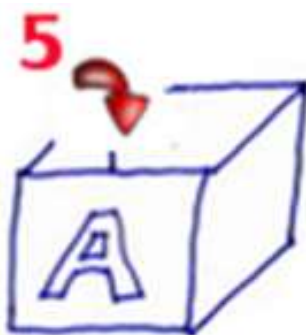
Representa un **contenedor** de datos cuyo valor puede **variar** durante la ejecución de un programa

Variable = Contenedor



Variable
llamada

A



Almacenamos
el valor 5
en la variable A



La variable A
"vale" 5



¿Cómo se escribe?

Variable

Sintaxis del Nombre de la Variable:

Para nombrar/etiquetar una variable utilizaremos un conjunto de caracteres (letras y números) comenzando con una letra, seguidos de cualquier cantidad de letras, números y guiones bajo.

No se pueden utilizar Espacios

Evitaremos vocales acentuadas y ñ.



Ejemplos variables: edad, miNombre, radio1, tempMaxima, tempMinima, unAño, saldo, promedio

Ejemplos que no corresponde a variables: 1año, es mayor, Temp Mínima



Tipo de dato de una Variable

Determina los valores que puede tomar una variable y las operaciones que pueden realizarse con ella:

Tipo de dato	Ejemplos de datos/valores	Algunas operaciones que permite realizar el tipo de dato
Entero	-25 45 0 11247	Suma, Resta, Multiplicación, División, Resto Comparación por mayor, igual, menor ...
Float, Real o decimal	-25.02 0.24 3.142 124.58	Suma, Resta, Multiplicación, División Comparación por mayor, igual, menor ...
Booleano o lógico	true false	or, and, not ...
String (cadena de caracteres entre comillas)	"hola mundo" "abcd" "124" "tengo 32 años"	Concatenación Comparación por mayor, igual, menor ...



Tipo de dato de una Variable

Reforcemos los siguientes tipo de datos:

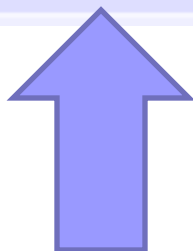
	12.1.00	
Booleano o lógico	true false	or, and, not ...
String (cadena de caracteres entre comillas)	"hola mundo" "abcd" "124" "tengo 32 años"	Concatenación Comparación por mayor, igual, menor ...



Tipo de dato de una Variable

Reforcemos los siguientes tipo de datos:

Booleano o lógico	true false	or, and, not ...
--------------------------	---------------	------------------



Una variable booleana*
Almacena uno de dos posibles
valores: verdadero o falso
(en general se usan los
nombres en inglés:
true, false)



Operadores
Booleanos o
lógicos

* O bien, variable de tipo de dato booleana



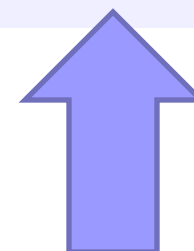
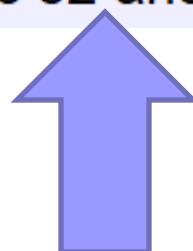
Tipo de dato de una Variable

Reforcemos los siguientes tipo de datos:

String (cadena de caracteres entre comillas)

“hola mundo”
“abcd”
“124”
“tengo 32 años”

Concatenación
Comparación por mayor, igual, menor ...



Una variable String* almacena cadena de caracteres (letras, nros, símbolos).

SIEMPRE utilizamos **comillas dobles** “ ” para indicar que se trata de una cadena de caracteres y no confundirlas con números, nombres de variables, valores booleanos o algún otro elemento del programa

Operadores
String

* O bien, variable de tipo de dato string



Tipo de dato de una Variable

Reforcemos los siguientes tipo de datos:

Booleano o lógico	true false	or, and, not ...
String (cadena de caracteres entre comillas)	"hola mundo" "abcd" "124" "tengo 32 años"	Concatenación Comparación por mayor, igual, menor ...

Atención! no confundir los siguientes valores :

true —————> Valor de tipo Booleano

"true" —————> Valor de tipo String (por las " ")

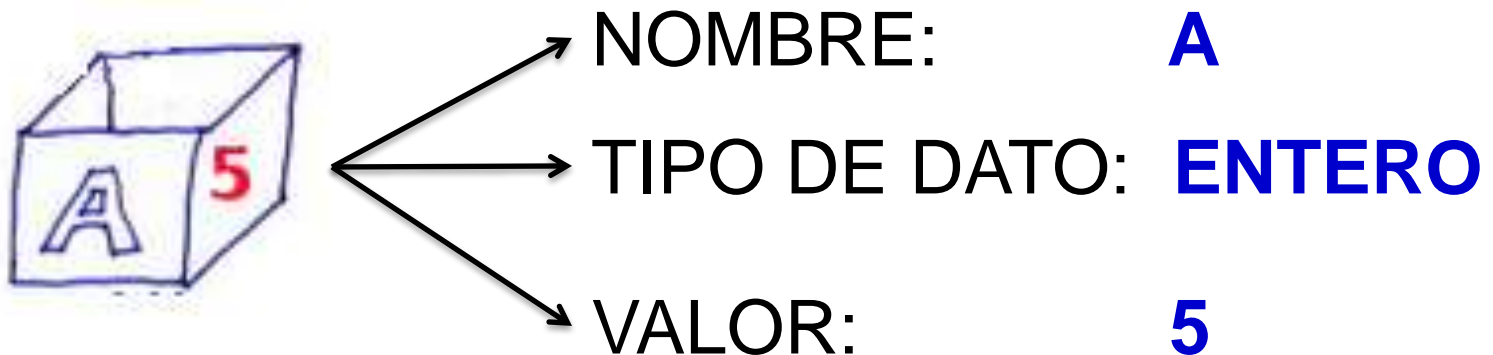




Variable

Resumiendo...

Nos referimos a una variable utilizando su **NOMBRE**.
Contendrá valores de un determinado **TIPO DE DATO**.
Almacena, guarda o contiene un **VALOR** o **DATO**





Lenguaje de Diseño: Pseudocódigo

Especificaremos las instrucciones de los algoritmos en un lenguaje de pseudocódigo.

Luego lo traduciremos al lenguaje de programación elegido (PHP).



Instrucciones

Orden clara y precisa para ser ejecutada por una computadora.



Instrucciones básicas:

- Instrucción de **Asignación**
- Instrucción de **Entrada**
- Instrucción de **Salida**

Cada instrucción implicará el uso de uno o varios componentes del hardware de una computadora



¿Qué es una Computadora?



HARDWARE : son los componentes físicos.

SOFTWARE : son los programas

aplicaciones de usuario

Sistema Operativo

Hardware

Programas:

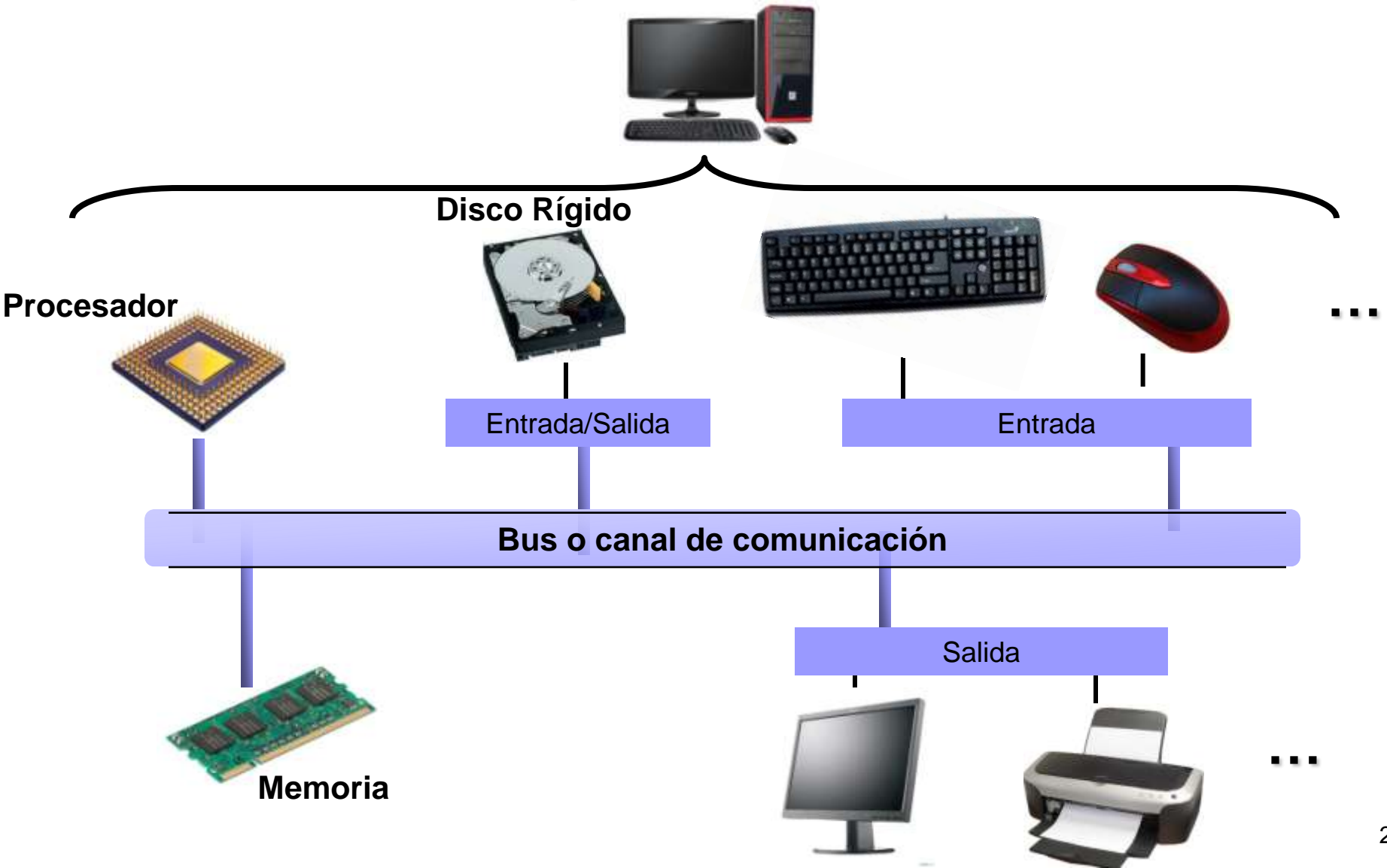
Conjunto de instrucciones
que resuelven un problema

Algunas capas del sistema de computadora



Computadora

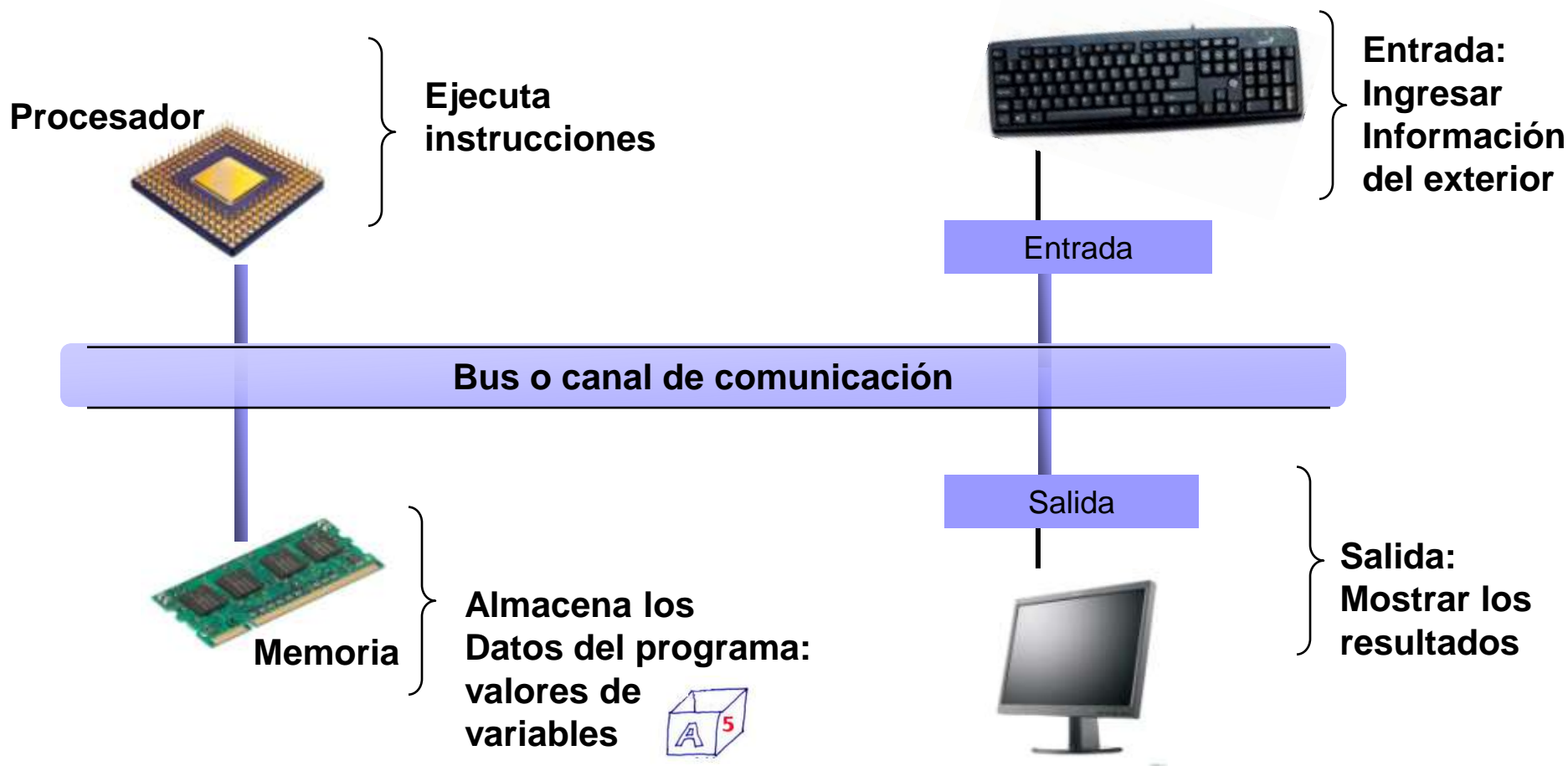
Componentes hardware





Computadora

Componentes hardware que utilizaremos los programas de la materia IP





Instrucciones

Orden clara y precisa para ser ejecutada por una computadora.



Instrucciones básicas:

- Instrucción de **Asignación**
- Instrucción de **Entrada**
- Instrucción de **Salida**

¿Qué tenemos que estudiar de cada una?

1º) Sintaxis (Cómo se especifica, cómo se escribe)

2º) Cómo se ejecuta? (Qué hace, qué efecto tiene en la computadora)



Instrucción de asignación

- Es la instrucción que nos permite **asociar un valor a una variable**, i.e., nos permite almacenar/guardar un dato en una variable.
- Su sintáxis es: **X ← E** donde X es una variable y E puede ser:
 - Un valor; ejemplos: `radio ← 13`
`figura ← 'circulo'`
 - Una expresión; ejemplo: `sup ← 3.14*radio*radio`
 - Otra variable que ya cuenta con un valor. ejemplo: `Z ← sup`
 - una función con retorno (tema de la unidad 4)
- Para ejecutar la instrucción tener en cuenta:
 - 1º) Obtener el valor de resolver la expresión E
 - 2º) Almacenar el valor en la variable



Instrucción de asignación

¡Importante!

Debe quedar claro que la asignación **no es una igualdad**. Implica **almacenar** un valor en una variable*.

Consideremos este ejemplo de algoritmo de 2 instrucciones:

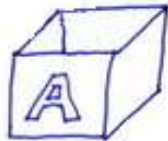
- 1 $A \leftarrow 5$
- 2 $A \leftarrow A - 3$

* Recordemos: una variable es un contenedor en la memoria de la computadora



1 $A \leftarrow 5$

2 $A \leftarrow A - 3$



Variable
llamada
A



Asignamos un 5
a la variable A
 $A \leftarrow 5$



La variable A
"vale" 5



Nombre: **A**

Valor: -

Tipo de Dato: -



Nombre: **A**

Valor: **5**

Tipo de Dato:

Entero/ Integer/ Int



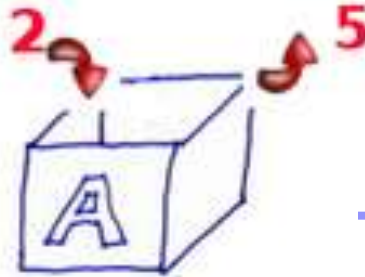
① $A \leftarrow 5$

② $A \leftarrow A - 3$



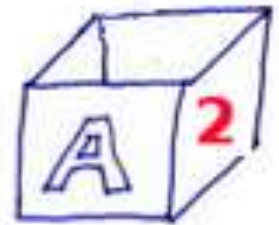
La variable A
"vale" 5

Resolver
la expresión
 $A - 3$, es decir,
 $5 - 3 = 2$



Asignamos un 2 a
la variable A
 $A \leftarrow 2$

A deja de "valer" 5
y pasa a "valer" 2
(de ahí que se
llame variable)



La variable A
"vale" 2

Nombre: A
Valor: 5
Tipo de Dato:
Entero/ Integer/ Int

Nombre: A
Valor: 2
Tipo de Dato:
Entero/ Integer/ Int



Instrucción de Entrada y Salida

Estas instrucciones permiten la comunicación con el mundo exterior:

- el ingreso de datos a un programa y
- la visualización de resultados.



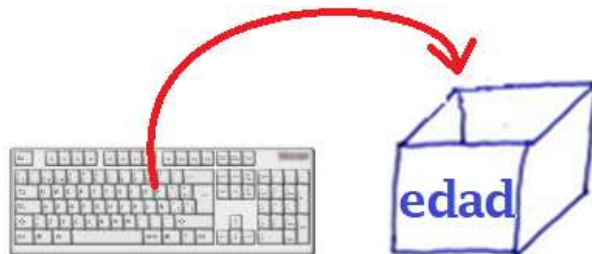
Instrucción de Entrada

Las instrucciones de entrada, al igual que la asignación, permiten asociar un valor a una variable, pero el valor es tomado desde el exterior.

La sintáxis correspondiente es:

LEER(edad)

edad es el nombre de una variable a la que deseamos asignarle un valor. El valor que será asociado a la variable ***edad*** podrá ser distinto cada vez que se ejecute esta instrucción.





Instrucción de Salida

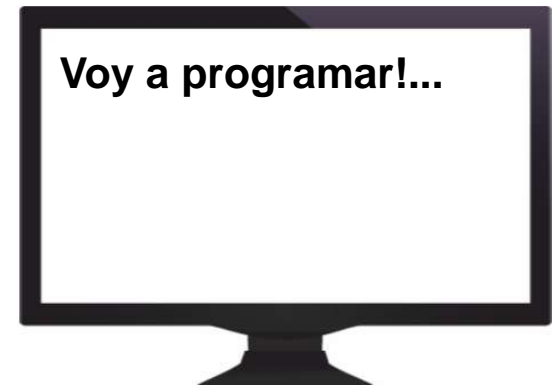
Las instrucciones de salida permiten mostrar los resultados obtenidos por nuestro programa.

Nos permiten conocer el valor de una variable.

La sintaxis es:

ESCRIBIR(E)

ESCRIBIR(E1, E2, ..., En)



E puede ser una variable entonces se imprime el valor guardado en la variable

E puede ser una expresión entonces se resuelve la expresión y se imprime el resultado de la expresión

E puede ser un valor literal entonces se imprime el valor



Luego de especificar un conjunto de instrucciones, ¿En qué orden deben ser ejecutadas?

la respuesta nos la dan
las **Estructuras de Control**



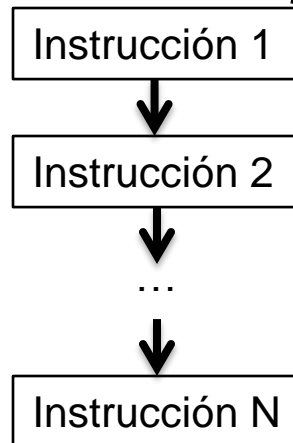
Estructuras de Control

Controlan el flujo de ejecución de las instrucciones de un programa.

Es decir, la estructura de control determina el orden en que se deben ejecutar las instrucciones

- **Estructura de Control Secuencial**

El programa se ejecuta instrucción por instrucción comenzando por la primera instrucción, de arriba hacia abajo (intuitivamente como una receta de cocina). Es decir, se ejecutan siguiendo una secuencia.



- **Estructura de Control Alternativa**

- **Estructura de Control Repetitiva**

} Unidades 5 y 6

Obs.: Nuestros primeros algoritmos respetarán una estructura de control secuencial ³⁸



Instrucciones

(Repasando)

Instrucción: orden o acción clara y precisa para ser ejecutada por una computadora.

- Instrucción de Asignación: $\text{edad} \leftarrow \text{edad} + 1$
- Instrucción de Entrada: $\text{LEER}(\text{nombre})$
- Instrucción de Salida: $\text{ESCRIBIR}(\text{suelo})$

¿Qué efecto tendrá la ejecución de cada instrucción en una computadora?





Ejemplo



Efecto de las instrucciones sobre los componentes de una computadora

PROGRAMA nacimiento

Nombre que
identifica al
algoritmo

(*Calcula el año de nac. de una persona *)

Comentario que
describe ¿qué
hace el algoritmo?
Se usan: (* *)

String nombre, **Entero** edad, anio, anioNac

ESCRIBIR("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio ← 2020

anioNac ← anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Declaración de
las variables que
se utilizan en el
algoritmo
(tipo de
dato y nombre)

FIN PROGRAMA



Efecto de las instrucciones sobre los componentes de una computadora

PROGRAMA nacimiento

(*Calcula el año de nac. de una persona *)

String nombre, **Entero** edad, anio, anioNac

ESCRIBIR("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio \leftarrow 2020

anioNac \leftarrow anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Especificación de las instrucciones que se ejecutarán secuencialmente (Estructura de control secuencial)

FIN PROGRAMA

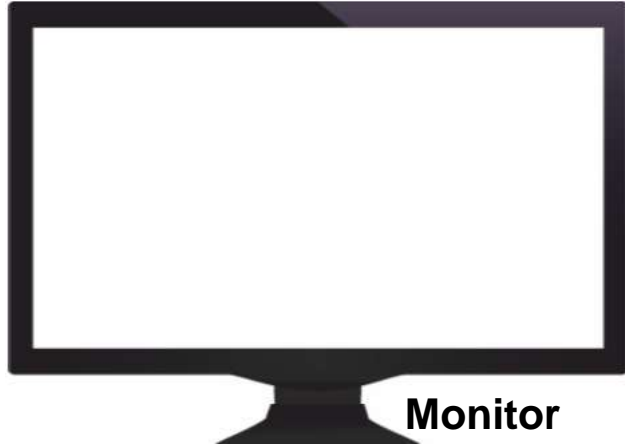


Estado inicial previo a ejecutar el algoritmo



Memoria → la representaremos como una tabla

nombre	edad	anio	anioNac



Monitor



Teclado



Instrucción:

1 **ESCRIBIR**("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio \leftarrow 2020

anioNac \leftarrow anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:



Ingrese su nombre:



nombre	edad	anio	anioNac





Instrucción:

2

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

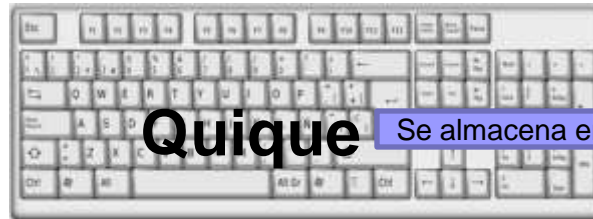
anio \leftarrow 2020

anioNac \leftarrow anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:

Ingrese su nombre:



Quique

Se almacena en

nombre	edad	anio	anioNac
Quique			





Instrucción:

3

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio \leftarrow 2020

anioNac \leftarrow anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:

Ingrese su nombre:
Ingrese edad:



nombre	edad	anio	anioNac
Quique			





Instrucción:

4

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio \leftarrow 2020

anioNac \leftarrow anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:



Ingrese su nombre:
Ingrese edad:



20

Se almacena en

nombre	edad	anio	anioNac
Quique	20		





Instrucción:

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

5

anio ← 2020

anioNac ← anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:

Ingrese su nombre:
Ingrese edad:



nombre	edad	anio	anioNac
Quique	20	2020	





Instrucción:

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio \leftarrow 2020

6

anioNac \leftarrow anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:

Ingrese su nombre:
Ingrese edad:



nombre	edad	anio	anioNac
Quique	20	2020	2020 -20



Atención! Escribir una expresión dentro de la variable es un error conceptual!
Recordar que una variable sólo puede almacenar un valor,
por lo tanto la tabla SÓLO puede contener valores NO se pueden escribir operaciones.
Si necesito hacer una cuenta lo hago en una hoja borrador



Instrucción:

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio ← 2020

6

anioNac ← anio - edad

ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:

Ingrese su nombre:
Ingrese edad:



nombre	edad	anio	anioNac
Quique	20	2020	2000





Instrucción:

Escribir ("Ingrese su nombre:")

LEER(nombre)

ESCRIBIR("Ingrese edad:")

LEER(edad)

anio \leftarrow 2020

anioNac \leftarrow anio - edad

7 ESCRIBIR (nombre, " naciste en ", anioNac)

Resultados de ejecutar la instrucción:



Ingrese su nombre:
Ingrese edad:
Quique naciste en 2020



nombre	edad	anio	anioNac
Quique	20	2020	2000





Traza

¡Muy importante!

¡Fundamental para aprender a Programar!

Nos ayuda a verificar
si el algoritmo es correcto



Traza de un algoritmo

Se puede definir como: la ejecución de las instrucciones que componen el algoritmo.

La traza de un algoritmo muestra el valor que van adoptando las variables a medida que se van ejecutando las instrucciones.

Es muy importante y útil para **testear el algoritmo especificado**, es decir, establecer si con un ejemplo de entrada obtenemos el resultado esperado.



¡Algoritmo para realizar la traza de un algoritmo!

Construcción de la tabla:



- 1) Determinar cuáles son todas las variables del programa
- 2) Armar una tabla con tantas columnas como variables distintas existan y nombrar cada columna con el nombre de una variable
- 3) Si existe una instrucción de Salida (ESCRIBIR), entonces agregar una columna “salida/pantalla”



¡Comenzamos a ejecutar!

Ejecución del algoritmo Secuencial:



- 4) Ejecutar la secuencia de instrucciones una a una, de arriba hacia abajo. Luego de ejecutar una instrucción puede pasar a ejecutar la siguiente (no puede saltar instrucciones!):
 - a) Si es una instrucción de lectura, LEER(A): guardar el valor ingresado por teclado en la columna A.
 - b) Si es una instrucción de escritura, ESCRIBIR(E): resolver la expresión E y escribir el resultado en la columna “salida/pantalla”
 - c) Si es una instrucción de Asignación $X \leftarrow E$: resolver la expresión E y escribir el valor en la columna nombrada X



¡Algoritmo para realizar la traza de un algoritmo!

Ejecución del algoritmo Secuencial:



**Atención! una variable sólo puede almacenar un valor,
por lo tanto la tabla SÓLO puede
contener valores
NO se pueden escribir operaciones**



nombre	edad	anio	anioNac	Salida / pantalla
Quique	20	2019	2019- 20	Ingrese su nombre: Ingrese edad: Quique naciste en 1998

Vs.

nombre	edad	anio	anioNac	Salida/pantalla
Quique	20	2019	1999	Ingrese su nombre: Ingrese edad: Quique naciste en 1998



Práctico 2 – Ejercicio 2-i

PROGRAMA principal

(* *)

Entero a, b, c, resultado

ESCRIBIR("Ingrese un número: ")

LEER(a)

ESCRIBIR("Ingrese un número: ")

LEER(b)

ESCRIBIR("Ingrese un número: ")

LEER(c)

resultado \leftarrow a + b + c

ESCRIBIR("El resultado es:", resultado)

FIN PROGRAMA



Práctico 2 – Ejercicio 2-i

PROGRAMA principal

```
(* *)
Entero a, b, c, resultado
ESCRIBIR("Ingrese un número: ")
LEER(a)
ESCRIBIR("Ingrese un número: ")
LEER(b)
ESCRIBIR("Ingrese un número: ")
LEER(c)
resultado  $\leftarrow$  a + b + c
ESCRIBIR("El resultado es:", resultado)
FIN PROGRAMA
```

Contestar:

1. Cuántas variables se definieron?
2. Cuantas Instrucciones de salida se especificaron?
3. Cuantas Instrucciones de entrada se especificaron?
4. Cuantas Instrucciones de asignación se especificaron?
5. Hacer la traza para los siguientes datos de entrada: 10, -80, 50. El dato de salida es:
6. Hacer la traza para siguientes datos de entrada: -4, -1, 25. El dato de salida es:
7. En el comentario (* *) explique coloquialmente qué hace el algoritmo:58



Práctico 2 – Ejercicio 2-i

PROGRAMA principal

(* *)

Entero a, b, c, resultado

① **ESCRIBIR**("Ingrese un número: ")

② **LEER**(a)

③ **ESCRIBIR**("Ingrese un número: ")

④ **LEER**(b)

⑤ **ESCRIBIR**("Ingrese un número: ")

⑥ **LEER**(c)

⑦ resultado \leftarrow a + b + c

⑧ **ESCRIBIR**("El resultado es:", resultado)

FIN PROGRAMA