Universidad Nacional de Jujuy
Facultad de Ingeniería

Introducción a la Programación
Programación I

Estructura secuencial

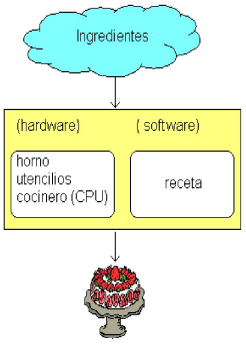
Contenido

- Introducción
- Concepto de algoritmo
- Resolución de problemas mediante algoritmos
- Teorema Fundamental de la Programación Estructurada.
- Ambiente del problema
- Expresiones
- Concepto de asignación
- Ejemplos de algoritmos secuenciales

Introducción

- Historia del cálculo
- Sistemas de numeración
- Historia de la computación
  - Generación de computadoras
  - Desarrollo del *hardware* y el *software*
- Internet
- Avances científicos y tecnológicos

Problema: Hacer un pastel



Ingredientes

- 1½ taza de harina
- 1 taza azúcar
- ½ cucharita de sal
- 1 cucharita de bicarbonato de sodio
- 3 cucharas de chocolate en polvo
- 1 cucharita de vinagre
- 6 cucharitas de aceite
- 1 cuchara de extracto de vainilla
- 1 taza de agua
- 1 huevo
- Decoración: dulce de leche, chocolate, praliné

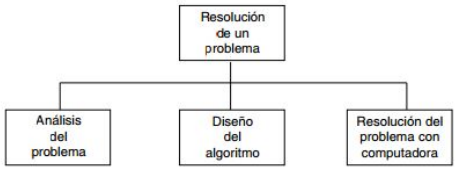
Receta: Hacer un pastel

- Mezclar los ingredientes secos.
- Mezclar los ingredientes líquidos.
- Enmantecar y enharinar un recipiente.
- Vertir la mezcla en el recipiente.
- Hornear a 177 °C. Por 30 minutos.
- Dejar que el pastel se enfríe.
- **Decorar*.**

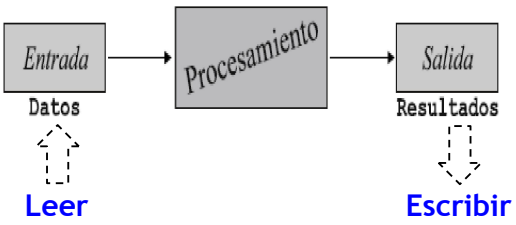
Receta: Decorar un pastel

- **Decorar*:**
  - Bañar con chocolate arriba y hacer telaraña de chocolate blanco.
  - Cubrir los costados con dulce de leche y pegarle praliné.
  - Terminar con manga de dulce de leche con pico rizado.

Resolución de un problema



Análisis del problema



Algoritmo

*Es un conjunto prescrito de **instrucciones** o reglas bien definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos, que no generen dudas a quien deba llevarlo a cabo.*

instrucción ≡ acción ≡ sentencia ≡ estructura

Características de un algoritmo

- ***Preciso**: orden de cada paso*
- ***Definido**: Si se realiza más de una vez se debe obtener el mismo resultado*
- ***Finito**: número finito de pasos*
- ***Acciones primitivas**: entendible para el que lo realiza*

Teorema Fundamental de la Programación Estructurada

Todo programa propio se puede escribir utilizando únicamente las siguientes estructuras de control:

- secuencial,
- selectiva,
- repetitiva

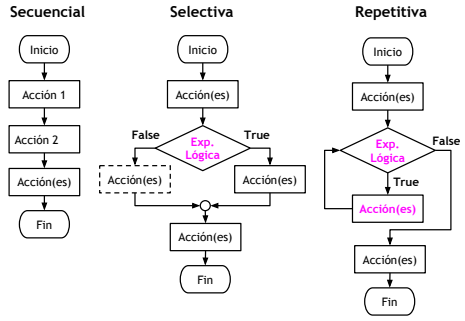
Teorema Fundamental de la Programación Estructurada

Un programa propio se define como propio si:

- Tiene un único punto de entrada y un único punto de salida.
- Existen caminos desde la entrada hasta la salida que pasan por todas las partes del programa.
- Todas las instrucciones son ejecutables y no existen bucles sin fin.

Böhm, G. Jacopini, 1966

Estructuras



Herramientas para el diseño de algoritmos

- Lápiz y papel
- Pseudocódigo
- Diagrama de flujo
- Diagrama Nassi-Shneiderman
- Lenguaje de programación (Python)
- Entorno integrado de desarrollo (VSC)

Ambiente del algoritmo

- **Identificador:** Nombre que no es propio del lenguaje, debe ser significativo.
  - Ej: temperatura, a-b, a_b, 1estudiante
- **Declaración**: nombre, tipo (en Python no se declara el tipo de las variables)
- **Constantes:** Tienen un valor fijo, que se le da cuando se define y no se modifica en el ambiente del programa.
  - Ej: PI = 3.14
- **Variables:** El valor puede cambiar

Estilo de programación

- camelCase, snake_case
- Ejemplos
  - Variables:
    - fechaIngreso, costoUnitario
    - fecha_ingreso, costo_unitario
  - Funciones
    - factorialRecursivo(), calcularPromedio()
    - factorial_recursivo(), calcular_promedio()
  - Constantes
    - IVA, INTERES_MAXIMO, PLAZO_MINIMO

Tipos de datos simples

Un tipo de dato establece qué valores puede tomar una variable y qué operaciones se pueden realizar sobre la misma.

Entero	Flotante	Lógico	Cadena
int 0, 1, 23, 3493 binario, octal, hexadecimal	float 0., 0.0, .0 1.0, 1.e0 2.99e-23 Números complejos	bol False, True	str ‘1’, “23”, “1.0”, ’UNJu’, “UBA”
+, -, *, /, ** Comparaciones	+, -, *, /, ** Comparaciones	Comparaciones	+ Comparaciones

Expresiones

Una expresión consta de un conjunto de Operandos y Operadores escritos de acuerdo a reglas predefinidas

Operando1 operador Operando2

Los **operandos** pueden ser:
- Variables
- Constantes
- Funciones

Los **operadores** pueden ser:
- Aritméticos
- Relacionales
- Lógicos

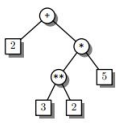
Operadores aritméticos

Operación	Operador	Aridad	Asociatividad	Precedencia
Exponenciación	**	Binario	Por la derecha	1
Identidad	+	Unario	—	2
Cambio de signo	-	Unario	—	2
Multiplicación	*	Binario	Por la izquierda	3
División	/	Binario	Por la izquierda	3
Módulo (o resto)	%	Binario	Por la izquierda	3
Suma	+	Binario	Por la izquierda	4
Resta	-	Binario	Por la izquierda	4

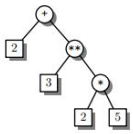
Tabla 2.1: Operadores para expresiones aritméticas. El nivel de precedencia 1 es el de mayor prioridad y el 4 el de menor.

Precedencia

2 + 3 ** 2 * 5



2 + 3 ** (2 * 5)



$V = \frac{4}{3} \prod \prod r^3$	**v = 4 / 3 * PI * r**3**
$\frac{\frac{b^2+a}{ac}}{\frac{3}{X+Y} - \frac{4.5-\sqrt{c}}{cos(6)}}$	**(b**2+a)/(a*c)/(3/(X+Y) -(4.5-c**0.5)/cos(6))**

Operadores relacionales

operador	comparación
==	es igual que
!=	es distinto de
<	es menor que
<=	es menor o igual que
>	es mayor que
>=	es mayor o igual que

Tabla 2.3: Operadores de comparación.

Ejemplos

(3 == 3) True

('A' > 'a') False

('UBA' > 'UNJu') False

(0.33 == 1/3) False

(True > False) True

Tabla de caracteres ASCII https://elcodigoascii.com.ar/

Asignación

variable ← Expresión

```
a ← 8
b ← 5 + 8.0
c ← 'UNJu' + '-' + 'FI'
d ← (0.0 = 0)
```

En Python

```
a = 8
b = 5 + 8.0
c = 'UNJu' + '-' + 'FI'
d = (0.0 == 0)
```


Conversión de tipos en Python

str(): Devuelve la representación en cadena de caracteres (string) del objeto que se pasa como parámetro.

int(): Devuelve un int a partir de un número o un string.

float(): Devuelve un float a partir de un número o un string

Ej: Algoritmo - estructura secuencial

x
3

$$1+\frac{x}{2}$$

```
Leer x
x ← 1+x/2
Escribir x
```

```
x = int(input('Ingrese x:'))
x = 1+x/2
print('Resultado x:', x)
```


Bibliografía

- Introducción a la Programación y a las Estructuras de datos. Braunstein y Gioia
- Fundamentos de la Programación. Joyanes Aguilar
- Introducción a la Programación con Python.
- https://docs.python.org/es/3/