


Programación I (Diagramación)

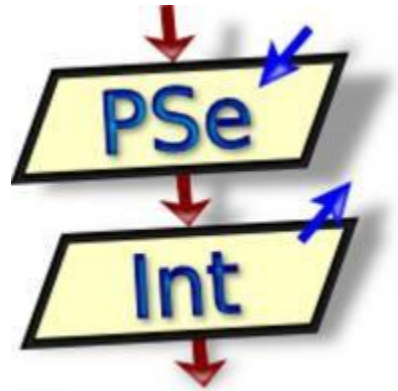
Docentes

- Ariel Tapia - Profesor
atapia@docentes.frgp.utn.edu.ar
- Walter Pintos- Jefe de TPs
walter.pintos@alumnos.frgp.utn.edu.ar

Contenidos

- Tipos de datos
 - Variables y constantes
 - Operadores
 - Estructura de secuencia
 - Estructura de decisión
 - Simple
 - Múltiple
 - Estructura de repetición
 - Ciclo exacto
 - Ciclo inexacto
 - Ciclos combinados
 - Vectores
 - Funciones
 - Proyectos de software
- 

Software



PSeInt



GitHub

Campus Virtual

Foros: Avisos, Dudas generales, Cafetería

Contrato didáctico

Cronograma

Guía de actividades

Repositorio de archivos

Videotutoriales



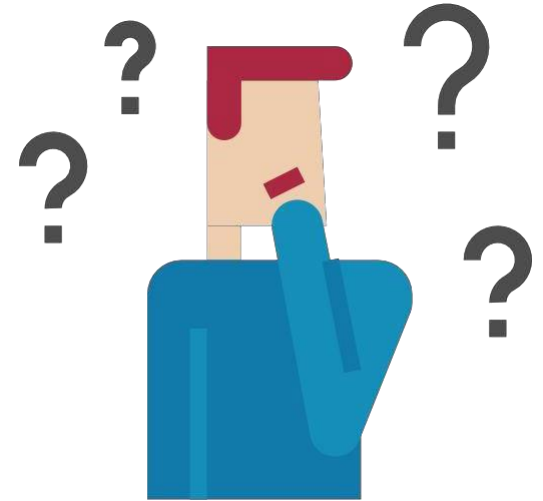
<https://frgp.cvg.utn.edu.ar/>

Usuario: [DNI.frgp](#)

Usuario: [DNI](#)

¿Qué es programar?

¿Es lo mismo programar que
codificar?



Proceso de desarrollo de un programa

1 Análisis del problema

Se analizan las características del problema. Se determinan los datos clave de entrada y salida. Se analizan requisitos y restricciones.

2 Diseño de la solución

Se determina en detalle las estructuras de datos y de programación que se utilizarán. Se hace un esquema general que resuelva claramente el problema.

3 Codificación de la solución en un lenguaje de programación

Se eligen las tecnologías más apropiadas para desarrollar la solución y se procede a codificarla.

4 Compilación

Dependiendo la tecnología, es necesario realizar un proceso de compilación para transformar el código en un programa ejecutable.

¿No compila?
Volver a (3)

5 Testing

Se ejecuta el programa creado y se comprueba su correcto funcionamiento.

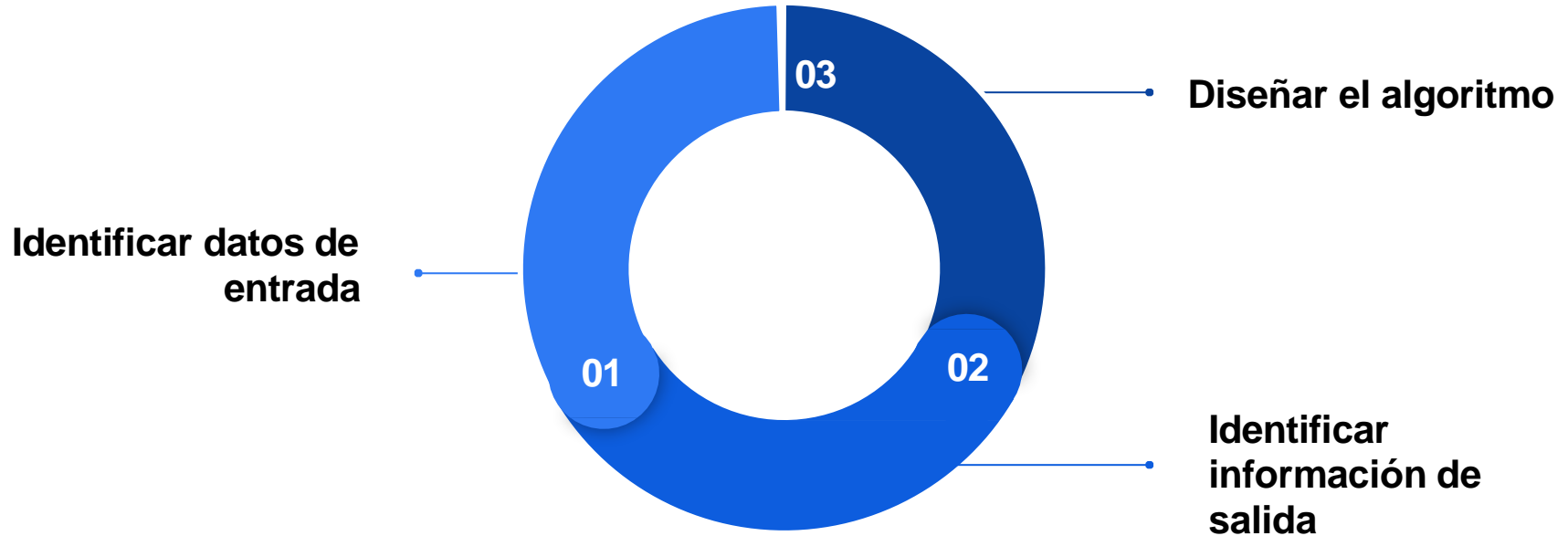
¿No funciona correctamente?
Volver a (2) ó (3)

6 Instalación y mantenimiento

De ser necesario, se realiza la instalación del programa, su puesta a punto y posterior mantenimiento de solución de bugs y mejoras.

¿No resuelve el problema?
Volver a (1)

Proceso de análisis de un problema



Proceso de resolución del problema

Datos de entrada

Determinar cuántos y cuáles son los datos de entrada de nuestro programa.

Ponerles un nombre y determinar su tipo de datos.

Proceso

El algoritmo ideado debe poder transformar los datos de entrada en la información de salida.

Se apreciará que el algoritmo sea eficiente en su resolución. Resolviendo el problema de la mejor manera y con el menor costo de recursos posibles.

Información de salida

La información de salida debe ser clara, prolija e informar estrictamente lo necesario.

Diseño de un algoritmo

Estructuras de datos

Variables simples o complejas, vectores, matrices, etc.



Estructuras de programación

Decisiones, estructuras de repetición, funciones, etc.



Esquema general del algoritmo

Secuencia de instrucciones no ambiguas y finitas que resuelven un problema en particular. El conjunto de instrucciones en un lenguaje determinado se llama código fuente.

Diseño de Algoritmos

■ Un algoritmo es una secuencia de pasos lógicos que permite dar solución a un problema específico.

La palabra algoritmo proviene del sobrenombre de un matemático árabe del siglo IX, Al-Khwārizmī, que fue reconocido por enunciar paso a paso las reglas para las operaciones matemáticas básicas con decimales (suma, resta, multiplicación y división).

Diseño de Algoritmos

Tipos de algoritmos:

- **No Gráficos:** Representa en forma descriptiva las operaciones que debe realizar un algoritmo (Pseudocódigo).

```
Inicio
    Leer sb, v1, v2, v3
    tot_vta = v1+v2+v3
    com = tot_vta * 0.10
    tpag = sb + com
    Escribir tpag, com
Fin
```

Diagrama de flujos

Un diagrama de flujo es la representación gráfica de un algoritmo. En realidad, muestra gráficamente los pasos o procesos a seguir para alcanzar la solución de un problema.

A continuación, **presentamos algunos de los símbolos que utilizaremos**, y una explicación de estos.

Diagrama de flujos.





SÍMBOLO	FUNCIÓN	DESCRIPCIÓN
	Inicio / Fin	Símbolo utilizado para marcar el Inicio y el Fin del diagrama de flujo.
	Entrada de Datos	Símbolo utilizado para Introducir los Datos de Entrada . Expresa Lectura. (Entrada)
	Salida de Datos	Símbolo utilizado para representar la Impresión de un resultado. Expresa Escritura. (Salida)
	Proceso	Símbolo utilizado para representar un Proceso . En su interior se expresan asignaciones, operaciones aritméticas, cambios de valor de celdas en memoria, etc.

Diagrama de flujos.

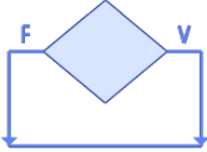
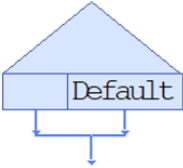


SÍMBOLO	FUNCIÓN	DESCRIPCIÓN
	Decisión	Símbolo utilizado para representar una Decisión . En su interior se almacena una condición, y dependiendo del resultado de la evaluación de esta, se sigue por una de las ramas o caminos alternativos. Este símbolo se utiliza en la Estructura Selectiva IF y IF-ELSE .
	Decisión múltiple	Símbolo utilizado para representar una Decisión Múltiple. En su Interior se almacena un selector, y dependiendo del valor de dicho selector se sigue por una de las ramas o caminos alternativos. Este símbolo se utiliza en la Estructura Selectiva múltiple SWITCH .
	Flujo de Datos	Símbolos utilizados para expresar la Dirección del Flujo del Diagrama .

Diagrama de flujos.

La correcta construcción de un diagrama de flujo es sumamente importante porque a partir del mismo se escribe un programa en algún lenguaje de programación. Si el diagrama de flujo está completo y correcto, el paso de este a un lenguaje de programación es relativamente simple y directo.


Variables

Representación simbólica de espacio de memoria. Es donde se almacenan los datos en procesamiento.

A solid blue triangle is located in the bottom right corner of the slide, pointing upwards and to the left.

Tipos de datos

Todos los datos tienen un tipo asociado con ellos. Un dato puede ser un simple carácter, tal como 'b', un valor entero tal como 35. El tipo de dato determina la naturaleza del conjunto de valores que puede tomar una variable.

A solid blue triangle is located in the bottom right corner of the slide, pointing upwards and to the left.


Tipos de datos

Tipos de datos	
Simples	Numéricos
	Lógicos
	alfanuméricos (String)
Estructurados	Arreglos (Vectores, Matrices)
	Registros
	Archivos
	Apuntadores

Tipos de datos

Datos numéricos: Permiten representar valores de forma numérica, esto incluye a los números enteros y los reales. Este tipo de datos permiten realizar operaciones aritméticas comunes.

Datos lógicos: Son aquellos que solo pueden tener dos valores (cierto o falso) ya que representan el resultado de una comparación entre otros datos (numéricos o alfanuméricos).

A solid blue triangle is located in the bottom right corner of the slide, pointing towards the top right.

Expresiones


Las expresiones son combinaciones de constantes, variables, símbolos de operación, paréntesis y nombres de funciones especiales.

$$A + (B + 3) / C$$

Expresiones

Cada expresión toma un valor que se determina tomando los valores de las variables y constantes implicadas y la ejecución de las operaciones indicadas.

Una expresión consta de operadores y operandos. Según sea el tipo de datos que manipulan, se clasifican las expresiones en:

- Aritméticas
 - Relacionales
 - Lógicas
- 
- A solid blue triangle is located in the bottom right corner of the slide, pointing towards the top right.


Operadores

Son elementos que relacionan de forma diferente, los valores de una o más variables y/o constantes. Es decir, los operadores nos permiten manipular valores.

Tipos de Operadores	Aritméticos
	Relacionales
	Lógicos

Operadores Aritméticos

Los operadores aritméticos permiten la realización de operaciones matemáticas con las variables. Los operadores aritméticos pueden ser utilizados con tipos de datos enteros o reales. Si ambos son enteros, el resultado es entero; si alguno de ellos es real, el resultado es real.

A solid blue triangle is located in the bottom right corner of the slide, pointing towards the top right.

Operadores Aritméticos

Operador	Descripción
Aritmético	
+	Suma
-	Resta
*	Multiplicación
/	División
% [MOD]	Módulo (Resto de la división entera)

Operadores Aritméticos


Ejemplo	Resultado
$4 + 2$	
$4 - 2$	
$7 * 2$	
$7 / 2$	
$7 \% 2$	

Operadores Aritméticos

Ejemplo	Resultado
$4 + 2$	6
$4 - 2$	2
$7 * 2$	14
$7 / 2$	3,5
$7 \% 2$	1

Operadores Aritméticos

Importante: En programación operador MOD se representa a través del símbolo %, no confundir con el símbolo de porcentaje que aparece en las calculadoras.

A solid blue triangle is located in the bottom right corner of the slide, pointing towards the top right.

Operadores Aritméticos

Todas las expresiones entre paréntesis se evalúan primero. Las expresiones con paréntesis anidados se evalúan dentro hacia fuera, el paréntesis más interno se evalúa primero.

Dentro de una misma expresión los operadores se evalúan en el siguiente orden:



Prioridad de los Operadores aritméticos

Prioridad	Operador	Nombre
1	$*$, $/$, $\%$	Multiplicación, división, modulo
2	$+$, $-$	Suma y resta

Los operadores en una misma expresión con igual nivel de prioridad se evalúan de izquierda a derecha.

Prioridad de los Operadores aritméticos

Expresión	Resultado
$4 + 2 * 5$	
$23 * 2 / 5$	
$3 + 5 * (10 - (2 + 4))$	
$0,35 + 5,09 - 14,0 / 40$	
$2,1 * (1,5 + 3,0 * 4,1)$	

Prioridad de los Operadores aritméticos

Expresión	Resultado
$4 + 2 * 5$	14
$23 * 2 / 5$	9,2
$3 + 5 * (10 - (2 + 4))$	23
$0,35 + 5,09 - 14,0 / 40$	5,09
$2,1 * (1,5 + 3,0 * 4,1)$	28,98

Operadores relacionales

Son necesarios para decisiones y ciclos. Nos permiten establecer proposiciones lógicas. El resultado de una proposición lógica puede ser **verdadero** o **falso**.

Operador	Significado
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que
==	Igual que
!=	Distinto que

Ejemplos

Si me pregunto `5 > b`, el resultado va a depender en función del valor de la variable `b`.

Por ejemplo:


Si `b` es igual a 6, el resultado es falso.

Si `b` es igual a 1, el resultado es verdadero.

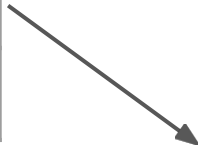
Operadores lógicos

Nos permiten combinar dos o más proposiciones lógicas

Operador		Significado
&&	and	Y lógico
	or	O lógico
!	not	Negación



A	B	A && B
verdadero	verdadero	verdadero
verdadero	falso	falso
falso	verdadero	falso
falso	falso	falso



A	B	A B
verdadero	verdadero	verdadero
verdadero	falso	verdadero
falso	verdadero	verdadero
falso	falso	falso

Operador de asignación

Nos permite asignar la expresión que se encuentra a la derecha del operador en la variable que se encuentra a la izquierda.



```
edad = 50;  
a = b;  
nombre = "Angel";  
caracter = 'X';  
precio = cant * pu;  
cont = cont + 1;
```



```
50 = edad;  
50 = 50;
```

Error de sintaxis

```
edad = edad;
```

Sintácticamente correcto pero sin sentido.

Ejercicio

Hacer un programa que permita ingresar dos números enteros por teclado. Luego calcular e informar la suma de ellos.

Ejercicio

Hacer un programa que permita ingresar dos números enteros por teclado. Luego calcular e informar la suma de ellos.

1

Datos de entrada:

2

Proceso:

3

Información de salida:

Ejercicio

Hacer un programa que permita ingresar dos números enteros por teclado. Luego calcular e informar la suma de ellos.

1

Datos de entrada: Dos números enteros.

2

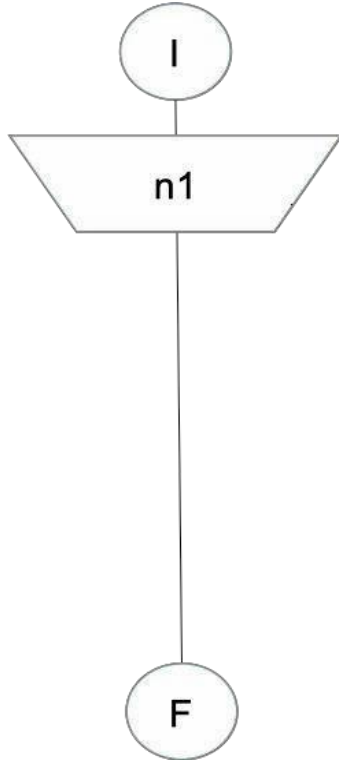
Proceso: Operación matemática de suma de ambos números.

3

Información de salida: Resultado de la suma

Resolución: Instrucciones

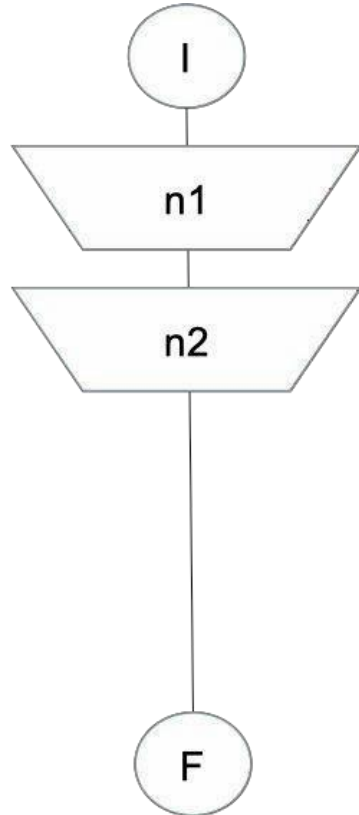
01



Ingresar el primer número

Resolución: Instrucciones

02

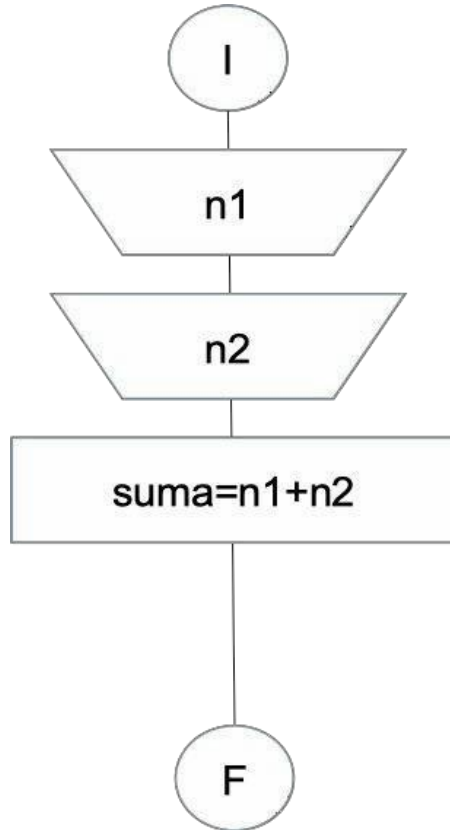


Ingresar el primer número

Ingresar el segundo número

Resolución: Instrucciones

03



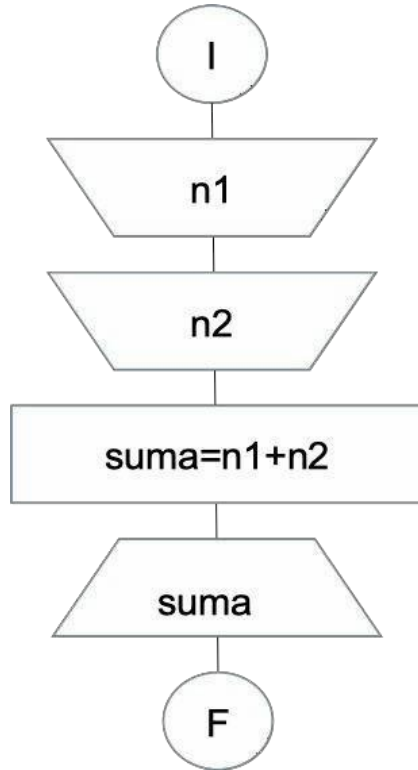
Ingresar el primer número

Ingresar el segundo número

Calcular la suma

Resolución: Instrucciones

04



■ Ingresar el primer número

■ Ingresar el segundo número

■ Calcular la suma

■ **Mostrar el resultado por pantalla**

Resolución: Código fuente

```
#include <iostream>
using namespace std;

int main(){
    int num1, num2, resultado;
    cin >> num1;
    cin >> num2;
    resultado = num1 + num2;
    cout << resultado;
    return 0;
}
```



¿¡Qué es todo eso!?

No se preocupen. Próximamente lo verán en las clases de codificación.

