

Programación I

Docentes

- Angel Simón - Profesor
- Brian Lara - Profesor
- Abel Faure - Jefe de trabajos prácticos
- Daniela Pinto - Jefe de trabajos prácticos
- Verónica Carbonari - Jefe de trabajos prácticos
- Martín García - Jefe de trabajos prácticos
- Leandro Carabajal - Ayudante de trabajos prácticos
- Marcos Mazzitelli - Ayudante de trabajos prácticos
- Germán Dieser - Ayudante de trabajos prácticos
- Josué Darío Solis - Ayudante de trabajos prácticos

Días de encuentros virtuales




Martes

18:00 a 20:30

Miércoles

19:30 a 22:00

Contenidos

- Variables y constantes
- Operadores
- Estructura de secuencia 
- Estructura de decisión
 - Simple
 - Múltiple
- Estructura de repetición
 - Ciclo exacto
 - Ciclo inexacto
 - Ciclos combinados
- Vectores
- Funciones
- Trabajo Integrador Final

Criterios de aprobación

Evaluación por procesos

Deben realizar una serie de actividades obligatorias durante la cursada para aprobar la materia.

- Aprobación de cuestionarios
- Participación activa en los foros
- Realización y defensa de un Trabajo Integrador Final

Herramientas para la resolución de problemas



Diagramación

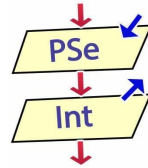
Mediante diagramas de flujo deberán poder ser capaces de representar la solución a la lógica de situaciones problemáticas.



Codificación

Mediante el lenguaje de programación C++ deberán poder ser capaces de realizar programas que brinden la solución a situaciones problemáticas.

Software



GitHub



Codificación: Codeblocks 20.03 (Windows y Linux)

Diagramación: PSeInt

Alternativas:

Codificación: Visual Studio Code + plugin C++ (Windows, Linux, Mac), Xcode (Mac)

Codificación: Zinjai (Windows), Visual Studio Community (Windows)

Diagramación: Draw.io, cualquier software para diagramar, papel y lápiz

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

Del código al programa

El proceso de transformar un **código fuente** en un **programa** que contiene las instrucciones que sean comprensibles por la computadora se llama **compilación**.



Codificación

Cada elemento que utilicemos en un **lenguaje de programación** debe estar sujeto a una estricta sintaxis. Los elementos que el lenguaje admite son:

programacion
Programacion
PROGRAMACION

Variables y constantes

Operadores

Expresiones

Palabras reservadas del lenguaje

Como muchos lenguajes, C y C++ son **case-sensitive**. Esto significa que hace diferencias entre mayúsculas y minúsculas.

Palabras reservadas del lenguaje

Palabras que el lenguaje utiliza para identificar tipos de datos, estructuras de programación, etc. Tienen un significado especial para el lenguaje y no pueden ser utilizados como **identificadores de nombre**.

Propósito	Palabras reservadas
Tipos de datos	bool, int, float, char, short, void, long, double
Elementos de programación	if, else, switch, default, break, for, while, do, return, auto, struct, class, static, virtual
Operadores	new, delete, sizeof

Tipos de datos

En nuestros programas será necesario poder representar elementos "de la vida real" tales como precios, nombres, edades, etc. Para ello contaremos con distintos tipos de datos para representar esta variedad de información:

Tipo de dato	Descripción
int ✓	Representa un número entero
float ✓	Representa un número con expresión decimal (real)
bool ✓	Representa un valor booleano (verdadero o falso)
char ✓	Representa un carácter (letra, número o símbolo)
string ✓	Representa un texto

Variables y constantes

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

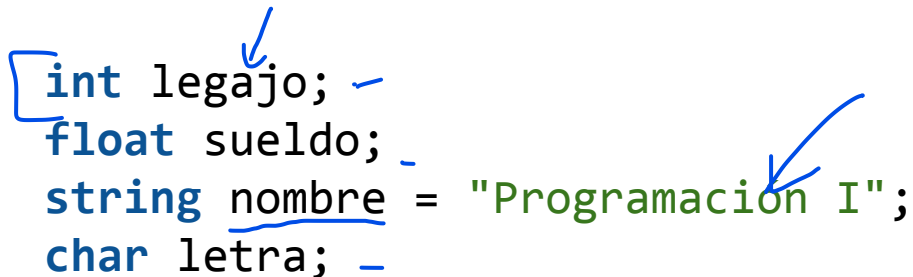
Una variable se identifica con un **tipo de dato** y un **identificador de nombre** (los elige el programador), y permite escribir un dato en la memoria o leer un dato de la memoria. Se puede modificar su valor las veces que sea necesario.

Una constante se identifica con la palabra reservada **const**, un tipo, un nombre (lo elige el programador), y un valor que no puede ser modificado durante el transcurso del programa en ejecución.

Variables y constantes

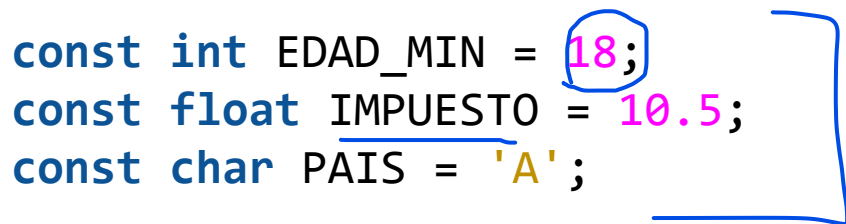
Ejemplos de declaración de variables

```
int legajo; -  
float sueldo; -  
string nombre = "Programación I";  
char letra; -
```



Ejemplos de declaración de constantes

```
const int EDAD_MIN = 18;  
const float IMPUESTO = 10.5;  
const char PAIS = 'A';
```



Operadores

Conjunto de símbolos y palabras reservadas que nos permiten hacer operaciones con expresiones.

Existen diferentes categorías de operadores:

■ **Asignación** ←

■ **Matemáticos** ✓

■ **Relacionales** ✓

■ **Lógicos** ✓

■ **De dirección e indirección** ✓

■ **Para memoria dinámica** ✓

■ **Etcétera**

Operadores matemáticos

Necesarios para realizar cálculos matemáticos. Los paréntesis tienen el mismo efecto que en la matemática en las expresiones algebraicas. Sin embargo, en programación no se utilizan corchetes ni llaves para la separación de términos.

Operador	Operación
+	Suma ✓
-	Resta ✓
*	Producto ✓
/	División real Cociente de la división entera
%	Resto de la división entera

Ejemplos

$$\begin{array}{l} 10 \mid 4 \\ ((2+3)*5)+10 \mid 35 \quad \begin{array}{l} 2 \quad 2 \text{ cociente} \\ \text{resto} \end{array} \\ 2+3*5+10 \mid 27 \\ 5 / 2 \quad \begin{array}{l} 2 \quad 2.5 \end{array} \\ 5.0 / 2 \quad 2.5 \\ 5 \% 2 \quad 1 \end{array}$$

Operadores matemáticos

Necesarios para realizar cálculos matemáticos. Los paréntesis tienen el mismo efecto que en la matemática en las expresiones algebraicas. Sin embargo, en programación no se utilizan corchetes ni llaves para la separación de términos.

Operador	Operación
+	Suma
-	Resta
*	Producto
/	División real Cociente de la división entera
%	Resto de la división entera

Ejemplos

$$((\cancel{2+3}) * 5) + 10 \rightarrow 35$$

$$\underbrace{2+3}_{5} * 5 + 10 \rightarrow 27$$

$$\underline{5 / 2} \rightarrow 2$$

$$5.0 / 2 \rightarrow 2.5$$

$$5 \% 2 \rightarrow 1$$

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;  
latitud = 45.22;  
nombre = "Angel";  
caracter = 'X';  
a = b;  
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.

✓ 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

Declarar las variables necesarias

```
int num1, num2, resultado;
```

02

Ingresa el primer número por teclado

```
cin >> num1;
```

Handwritten: *10* (with arrow pointing to `num1`)

03

Ingresa el segundo número por teclado

```
cin >> num2;
```

Handwritten: *25* (with arrow pointing to `num2`)

04

Realizar la suma

```
resultado = num1 + num2;
```

Handwritten: *10* above `num1`, *25* above `num2`

05

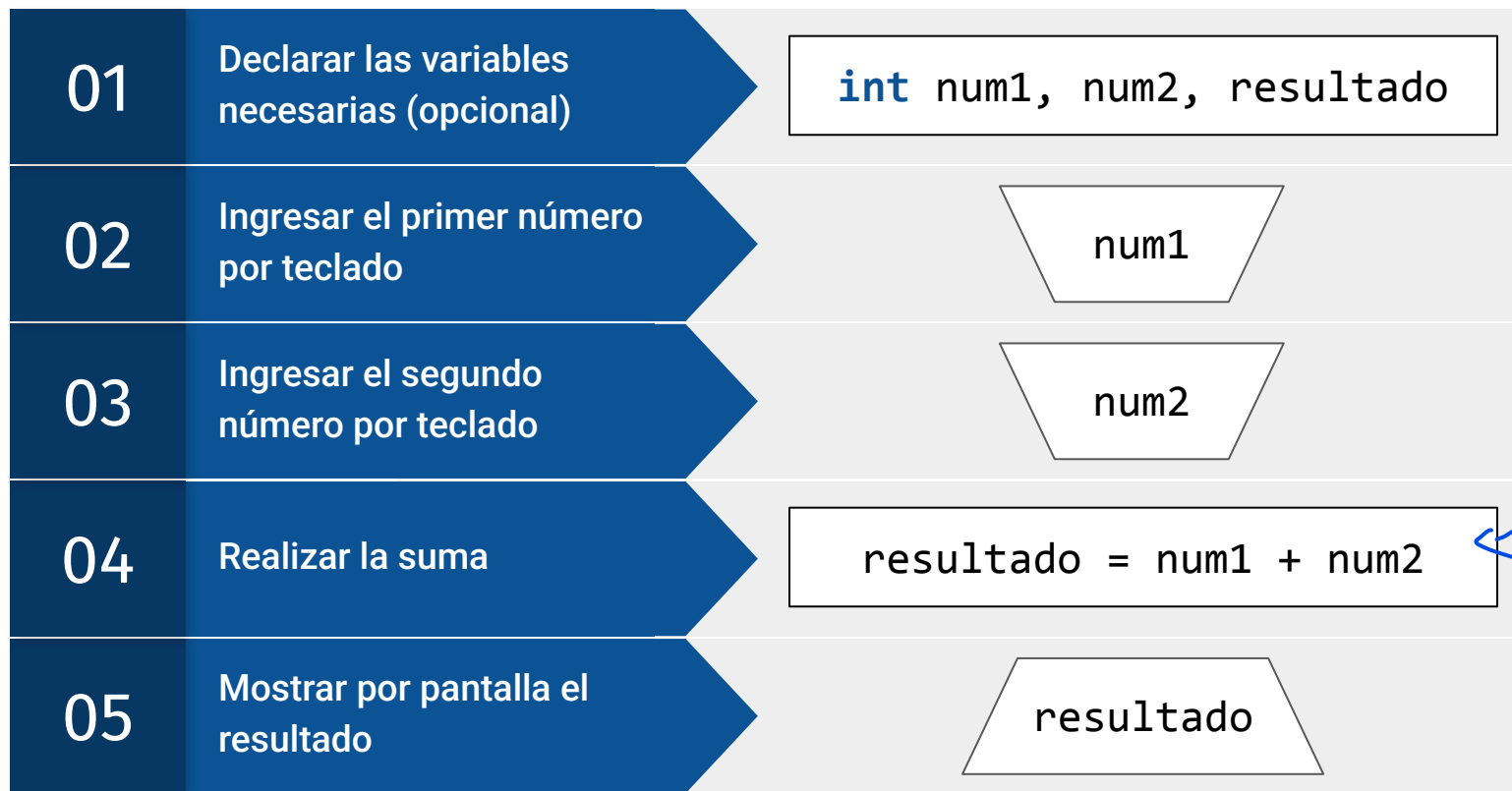
Mostrar por pantalla el resultado

```
cout << resultado;
```

Handwritten: *35* above `resultado`



Resolución: Instrucciones en diagrama



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. En breve lo codificamos en un IDE y lo mostramos paso a paso.



Resolución en C/C++



Créditos

- Logo de CodeBlocks: captura de pantalla de la aplicación
- Logo de Github: Derechos reservados Github.com
- Logo de C++: autor Jeremy Kratz
- Logo de VS Code: autor Microsoft