

# Programación 1

## Tema 1

---

Problemas de tratamiento de información,  
algoritmos y programas



Escuela de  
Ingeniería y Arquitectura  
**Universidad** Zaragoza

## Información sobre protección de datos de carácter personal en el tratamiento de gestión de grabaciones de docencia

Sesión con grabación



**Tratamiento:** Gestión de grabaciones de docencia

**Finalidad:** Grabación y tratamiento audiovisual de docencia y su evaluación

**Base Jurídica:** Art. 6.1.b), c) y d) Reglamento General de Protección de Datos

**Responsable:** Universidad de Zaragoza.

**Ejercicio de Derechos** de acceso, rectificación, supresión, portabilidad, limitación u oposición al tratamiento ante el gerente de la Universidad conforme a <https://protecciondatos.unizar.es/procedimiento-seguir>

**Información completa en:**

[https://protecciondatos.unizar.es/sites/protecciondatos.unizar.es/files/user/s/lopd/gdocencia\\_extensa.pdf](https://protecciondatos.unizar.es/sites/protecciondatos.unizar.es/files/user/s/lopd/gdocencia_extensa.pdf)

**Propiedad intelectual:** Queda prohibida la difusión, distribución o divulgación de la grabación y particularmente su compartición en redes sociales o servicios dedicados a compartir apuntes. La infracción de esta prohibición puede generar responsabilidad disciplinaria, administrativa y de índole civil o penal.

Fuente de las imágenes: <https://pixabay.com/es>



## Información sobre protección de datos de carácter personal en el tratamiento de gestión de grabaciones de docencia

---

- Se recuerda que la grabación de las clases por medios distintos a los usados por el profesor o por personas diferentes al profesor sin su autorización expresa no está permitida, al igual que la difusión de esas imágenes o audios.



# Problemas, algoritmos y programas

---

- Problemas de **tratamiento de información**
  - Objetivo: resolución **automática** del problema
  - ¿Quién? Un computador
  - Necesidad de **programarlo**

# Algoritmo

---

## □ Conjunto de operaciones

- ordenado,
- finito,
- carente de ambigüedades,

que permite hallar la solución de un problema [de tratamiento de información]

# Receta de tortilla de patata

## □ Ingredientes para 4 comensales

- 4 huevos
- Medio kilo de patatas
- Media cebolla
- Aceite de oliva
- Sal

## □ Elaboración:

- Corte las patatas en trocitos bien finos. Ponga a calentar abundante aceite de oliva en la sartén. Ponga las patatas en la sartén cuando el aceite esté bien caliente (nunca debe humear). Añada un poco de sal. Si la quiere con cebolla, añada la cebolla picada. Cuando las patatas estén bien doraditas, sáquelas y escúrralas. Bata bien los huevos, con una pizca de sal. Añada las patatas ya fritas y mezcle bien. Retire el aceite sobrante de la sartén y vuelva a ponerla al fuego. Cuando la sartén esté bien caliente, eche la mezcla de huevo y patatas. Cuando ya está hecha o cuajada por debajo, darle la vuelta con un plato plano o una tapadera.



# Índice

---

- ❑ Problemas de tratamiento de información
- ❑ Algoritmos y programas
- ❑ Ejemplos de programas C++
- ❑ Propiedades de un algoritmo



# Ejemplos de problemas de tratamiento de información

---

- ❑ Facilitar la escritura, edición, impresión y preservación digital de un texto
- ❑ Gestionar la información académica de los alumnos de la Universidad de Zaragoza
- ❑ Averiguar el número primo que sigue a 104743
- ❑ Permitir que una o varias personas jueguen en un entorno virtual persiguiendo un determinado objetivo
- ❑ Guiar el rayo láser que realiza queratectomía fotorrefractiva para corregir la miopía en ojos humanos
- ❑ Permitir que varias personas compartan entre sí en Internet información personal como noticias, fotografías, etc.



# Problemas, algoritmos y programas

---

- **Problema** (de tratamiento de información)



- **Método para su resolución**



- **Algoritmo**



- **Programa**

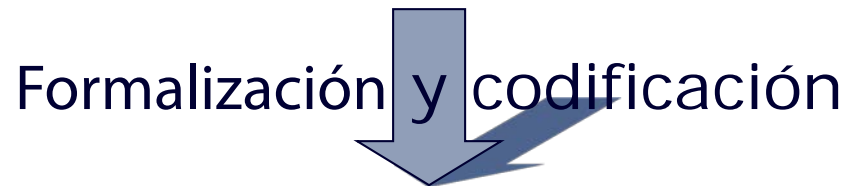
# Problemas, algoritmos y programas

---

- **Problema** (de tratamiento de información)



- **Método para su resolución**



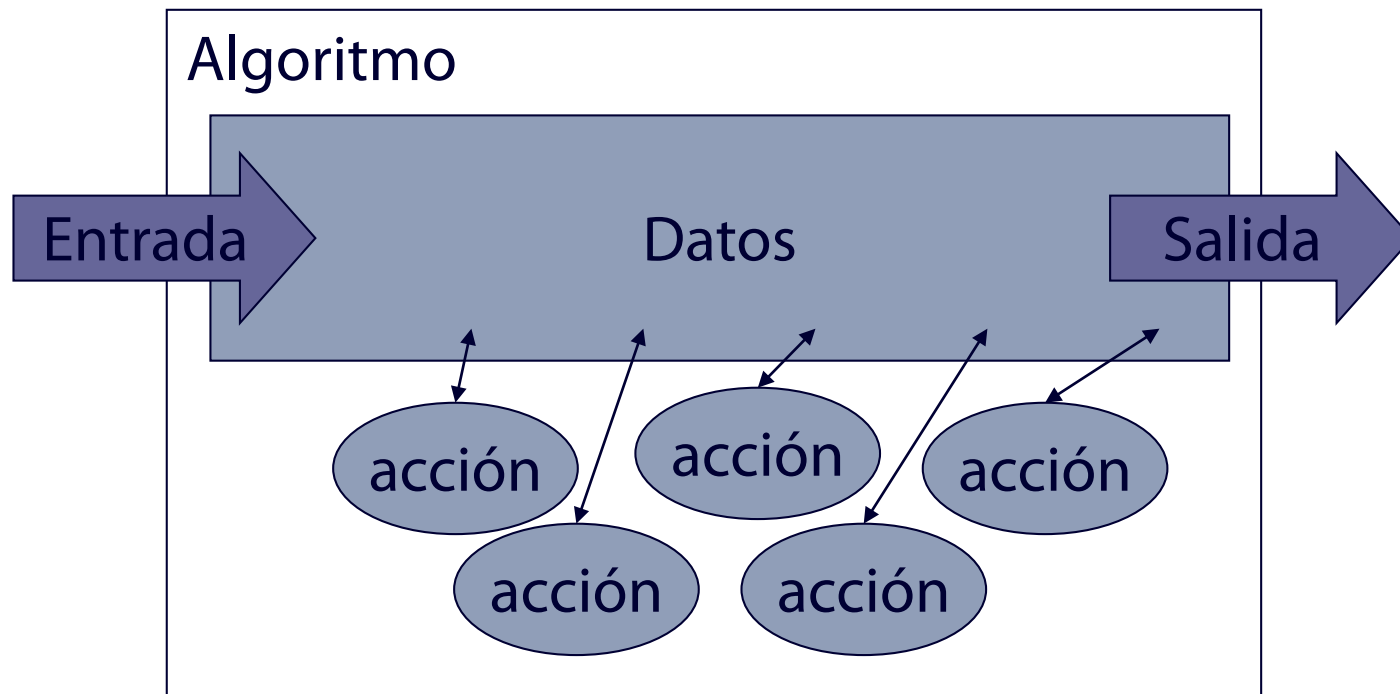
- **Programa**

# Algoritmo

---

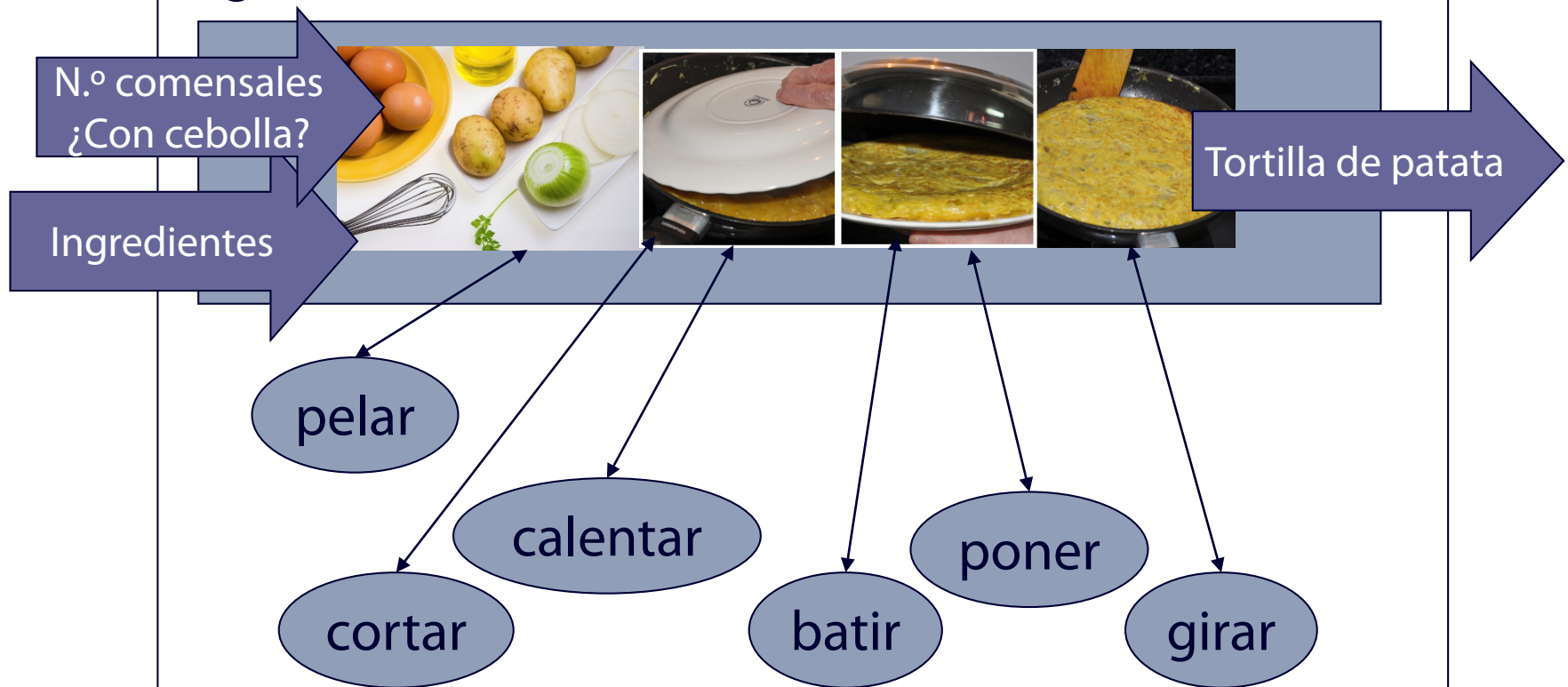
- Conjunto ordenado y finito de operaciones, carente de ambigüedades, que permite hallar la solución de un problema de tratamiento de información
- Consta de
  - **Descripción de la información** asociada al problema
  - **Descripción del modo de tratamiento** de esta información.

# Esquema de algoritmo



# Algoritmo para cocinar una tortilla de patata

## Algoritmo tortillaDePatata





# Conjunto de operaciones.

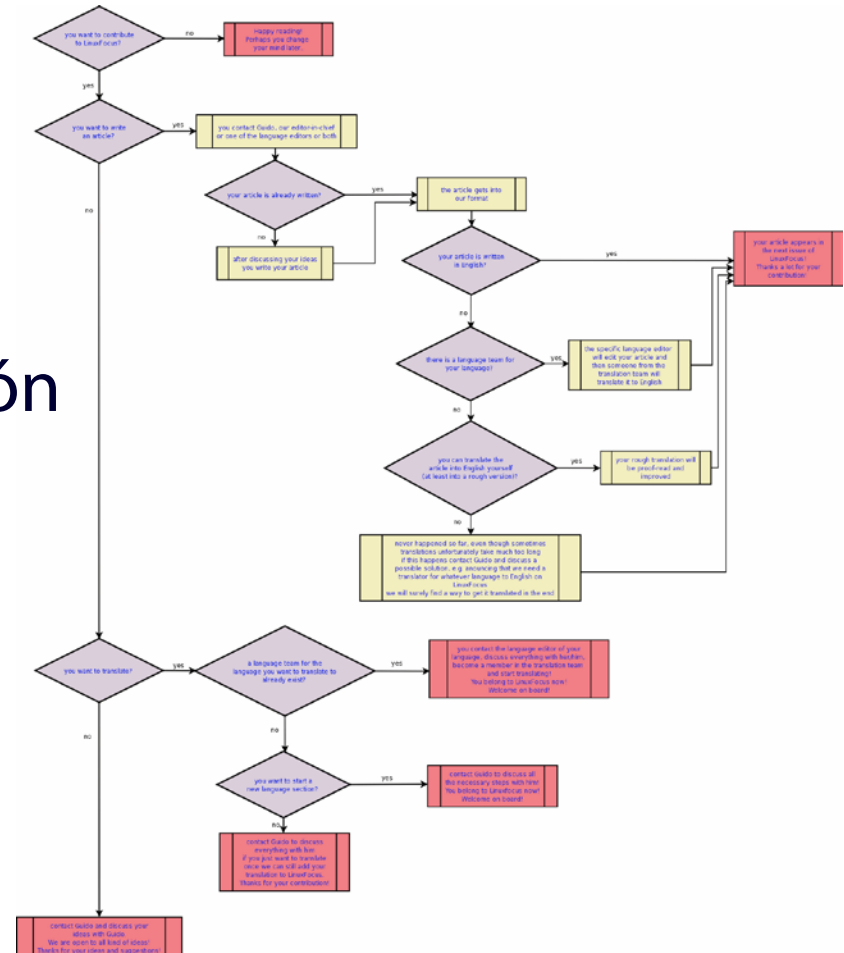
## Ejemplo de la tortilla de patata

---

- ☐ **Pelar** una *hortaliza*
- ☐ **Cortar** una *hortaliza*
- ☐ **Picar** una *hortaliza*
- ☐ **Poner** en un *recipiente* un *ingrediente*
- ☐ **Sacar** de un *recipiente* un *ingrediente*
- ☐ **Poner al fuego** un *recipiente*
- ☐ **Retirar del fuego** un *recipiente*
- ☐ **Escurrir** un *ingrediente*
- ☐ **Batir** un *ingrediente*
- ☐ **Girar** un *ingrediente*
- ☐ **Esperar**

# Expresión de un algoritmo

- Lenguaje natural
- Notación algorítmica
- Notación gráfica
  - Diagramas de flujo
- Lenguaje de programación
  - Ada, Pascal, Módulo-2, C
  - **C++**, Java
  - Python, Lisp, Prolog
  - Fortran, Cobol



# Algoritmo en lenguaje natural

## □ Ingredientes para 4 comensales

- 4 huevos
- Medio kilo de patatas
- Media cebolla
- Aceite de oliva
- Sal

## □ Elaboración:

- Corte las patatas en trocitos bien finos. Ponga a calentar abundante aceite de oliva en la sartén. Ponga las patatas en la sartén cuando el aceite esté bien caliente (nunca debe humear). Añada un poco de sal. Si la quiere con cebolla, añada la cebolla picada. Cuando las patatas estén bien doraditas, sáquelas y escúrralas. Bata bien los huevos, con una pizca de sal. Añada las patatas ya fritas y mezcle bien. Retire el aceite sobrante de la sartén y vuelva a ponerla al fuego. Cuando la sartén esté bien caliente, eche la mezcla de huevo y patatas. Cuando ya está hecha o cuajada por debajo, darle la vuelta con un plato plano o una tapadera.





(1)  $\frac{1}{2}$  Kg patatas en tiritas

1 cebolla picadita

sal

mezclar

RELLENO

aceite en sartén (  $\frac{1}{2}$  cm de espesor )

calentar a  $\frac{1}{2}$  fuego

(2) 20 min

rehogar a fuego medio, tapando la sartén. Mezclar relleno

### INGREDIENTES

$\frac{1}{2}$  kg. patatas

4 huevos

1 cebolla

250 cc aceite

sal

patatas cocidas

sacar con espumadera

PATATA Y CEBOLLA REHOGADA

aceite en la sartén

quitar aceite volcando sartén

(4)

aceite cubriendo justamente fondo sartén

MEZCLA REHOGADA

extender mezcla en fondo sartén

(5) 10 min

freir a medio fuego hasta cuajar tortilla por abajo

dar  $\frac{1}{2}$  vuelta a tortilla ( ayudarse de una tapadera )

4 huevos en bol

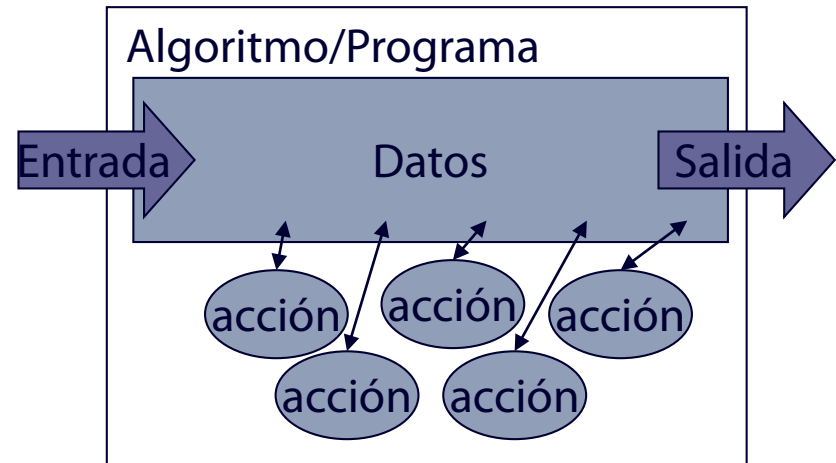
sal

batir bien

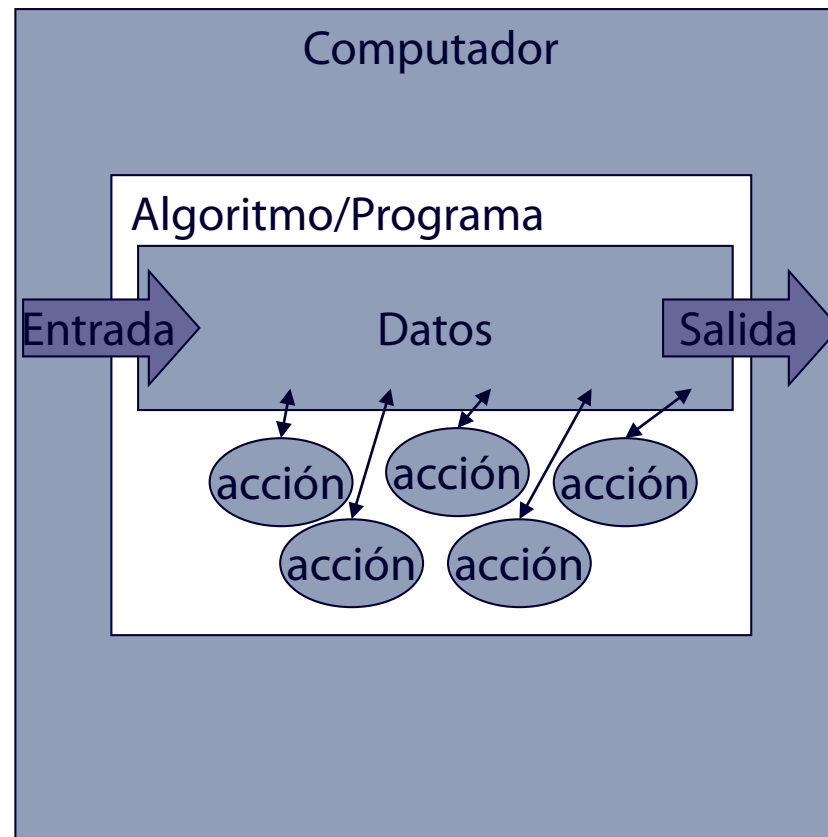
mezclar bien

# Nuestro modelo de computador

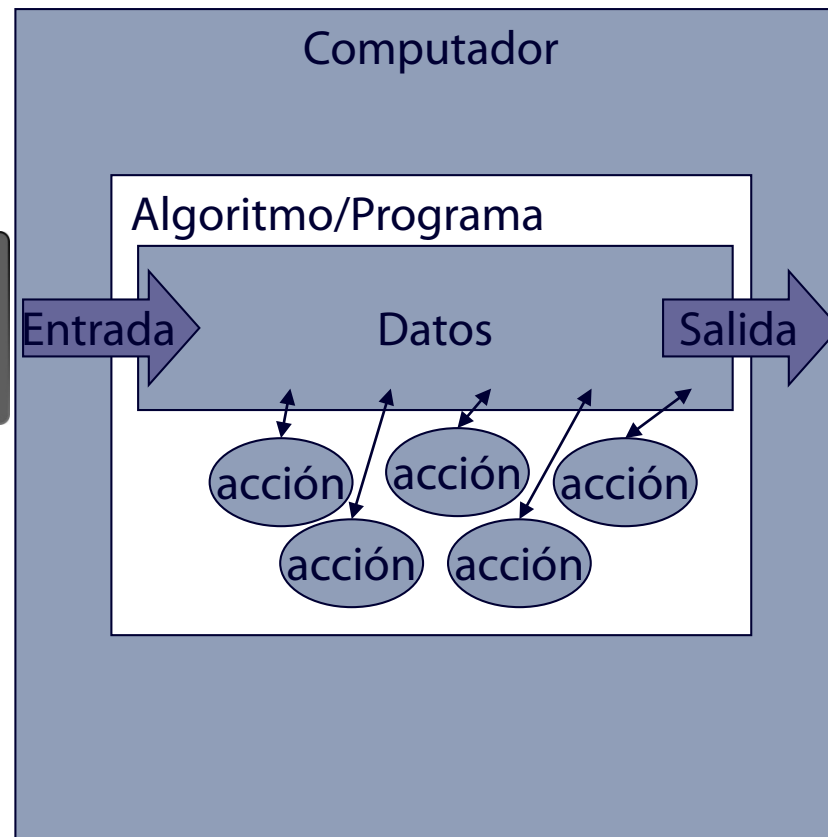
Computador



# Nuestro modelo de computador



# Nuestro modelo de computador



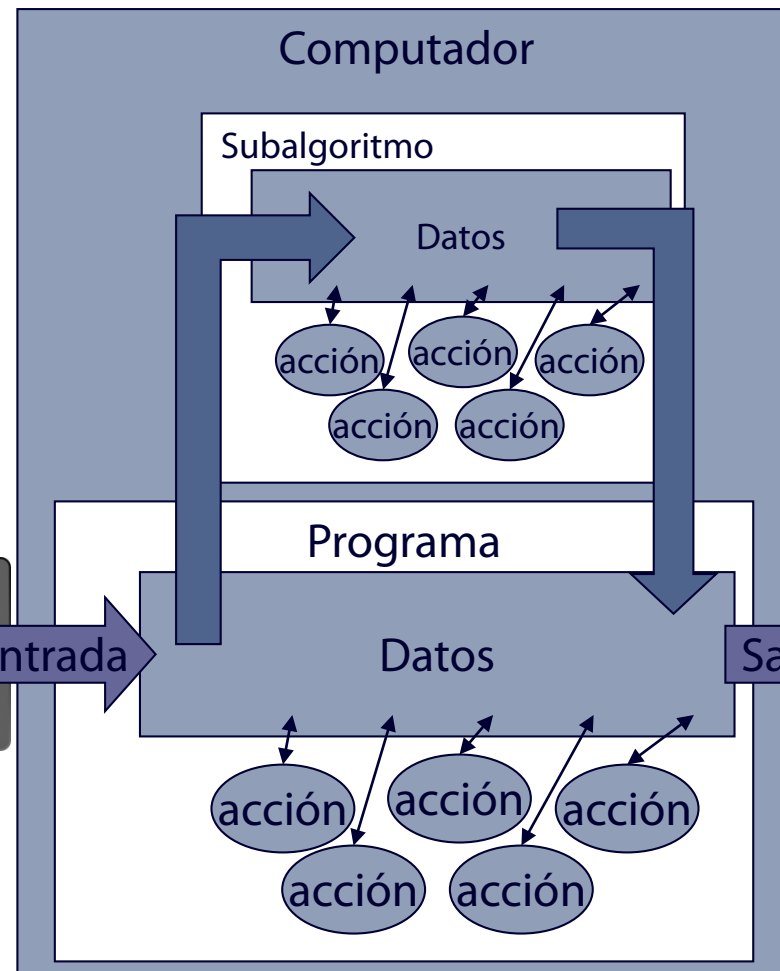


# Conjunto de operaciones

---

- ☐ **Realizar una operación aritmética** entre datos numéricos
- ☐ **Dar valor** a un dato
- ☐ **Modificar el valor** de un dato
- ☐ **Invocar** a otro algoritmo
- ☐ **Ejecutar opcionalmente** una acción
- ☐ **Repetir la ejecución** de una acción
- ☐ **Escribir** un dato en la pantalla
- ☐ **Leer** un dato del teclado
- ☐ ...

# Nuestro modelo de computador





# Un primer programa en varios lenguajes

---





# Un programa en Python

```
# Acción a ejecutar:  
print ('Bienvenidos a UNIZAR')
```



# Un algoritmo en una notación algorítmica

**algoritmo** bienvenida;

*{ Algoritmo que escribe una línea de texto en  
la pantalla con un mensaje de bienvenida }*

**principio**

*{ Acciones a ejecutar cuando sea invocado }*

escribir(pantalla, "Bienvenidos a UNIZAR");

**fin.**

# Un programa en Ada

```
with ada.text_IO;  
  
procedure bienvenida is  
  -- Programa que escribe en la pantalla una línea  
  -- de texto con un mensaje de bienvenida  
  
begin  
  -- Acciones que ejecutará el programa cada vez  
  -- que sea invocado  
  ada.text_IO.put("Bienvenidos a UNIZAR");  
  ada.text_IO.new_line;  
end bienvenida;
```

# Un programa en Java

```
package es.unizar.eina.prog1.cap1;

/**
 * Al construir un programa Java alrededor de esta clase se
 * ejecuta su método «main» que escribe un mensaje de
 * bienvenida a la Universidad.
 */
public class Bienvenida {

    /**
     * Escribe en la pantalla una línea de texto con el mensaje
     * “Bienvenidos a la Universidad”
     */
    public static void main(String[] argumentos) {
        // El código a ejecutar se limita a una sola instrucción
        System.out.println("Bienvenidos a UNIZAR");
    }
}
```

# Un primer programa en C++

```
#include <iostream>

/*
 * Programa que escribe en la pantalla el mensaje
 * "Bienvenidos a la Universidad"
 */
int main() {
    // primera instrucción
    std::cout << "Bienvenidos a la Universidad" << std::endl;

    // segunda instrucción
    return 0;
}
```

# ¿Cómo se ejecuta el código C++?

## □ Edición del código fuente

```
#include <iostream>

/*
 * Pre: ---
 * Post: Escribe por pantalla el mensaje
 */
int main() {
    // primera instrucción
    std::cout << "Bienvenidos a la Uni

    // segunda instrucción
    return 0;
}
```

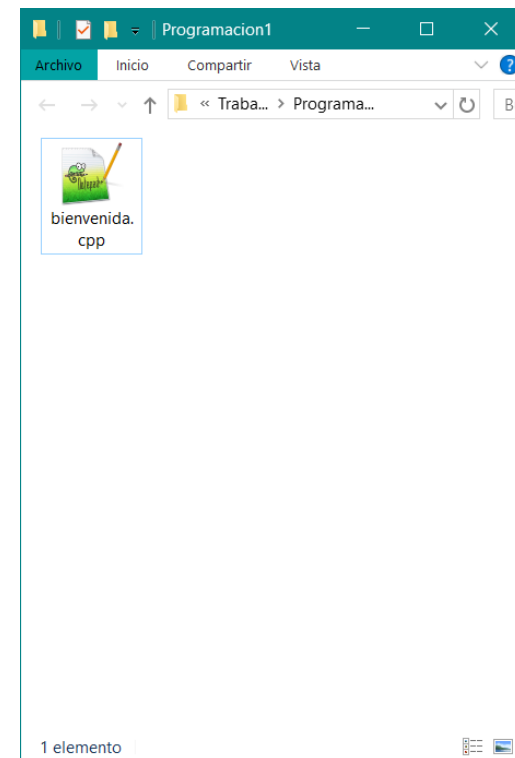
```
D:\bienvenida.cpp - Notepad++
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Macro  Ejecutar  TextFX  Plugins  Ventana  ?
bienvenida.cpp x
1  #include <iostream>
2
3  /*
4   * Pre: ---
5   * Post: Escribe por pantalla el mensaje
6   *      "Bienvenidos a la Universidad"
7   */
8  int main() {
9      // primera instrucción
10     std::cout << "Bienvenidos a la Universidad" << std::endl;
11
12     // segunda instrucción
13     return 0;
14 }
```

length : 277 lines : 14    Ln : 1 Col : 1 Sel : 0 | 0    Dos\Windows    UTF-8 w/o BOM    INS



# ¿Cómo se ejecuta el código C++?

- Edición del código fuente



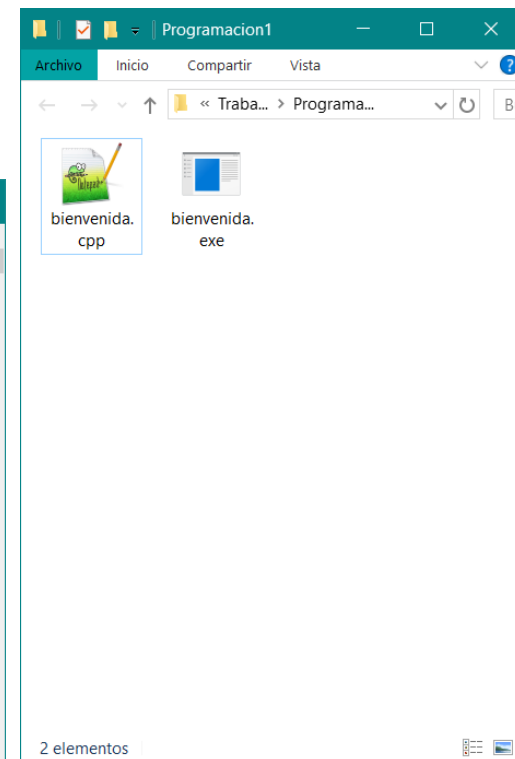
# ¿Cómo se ejecuta el código C++?

- ❑ Edición del código fuente
- ❑ Compilación del código fuente

```
C:\Windows\System32\cmd.exe

E:\Miguel\Documentos\Trabajo\Programacion1>g++ -o bienvenida.exe bienvenida.cpp

E:\Miguel\Documentos\Trabajo\Programacion1>
```



# ¿Cómo se ejecuta el código C++?

---

- ❑ Edición del código fuente
- ❑ Compilación del código fuente
- ❑ Ejecución del código ejecutable

```
C:\Windows\System32\cmd.exe

E:\Miguel\Documentos\Trabajo\Programacion1>g++ -o bienvenida.exe bienvenida.cpp

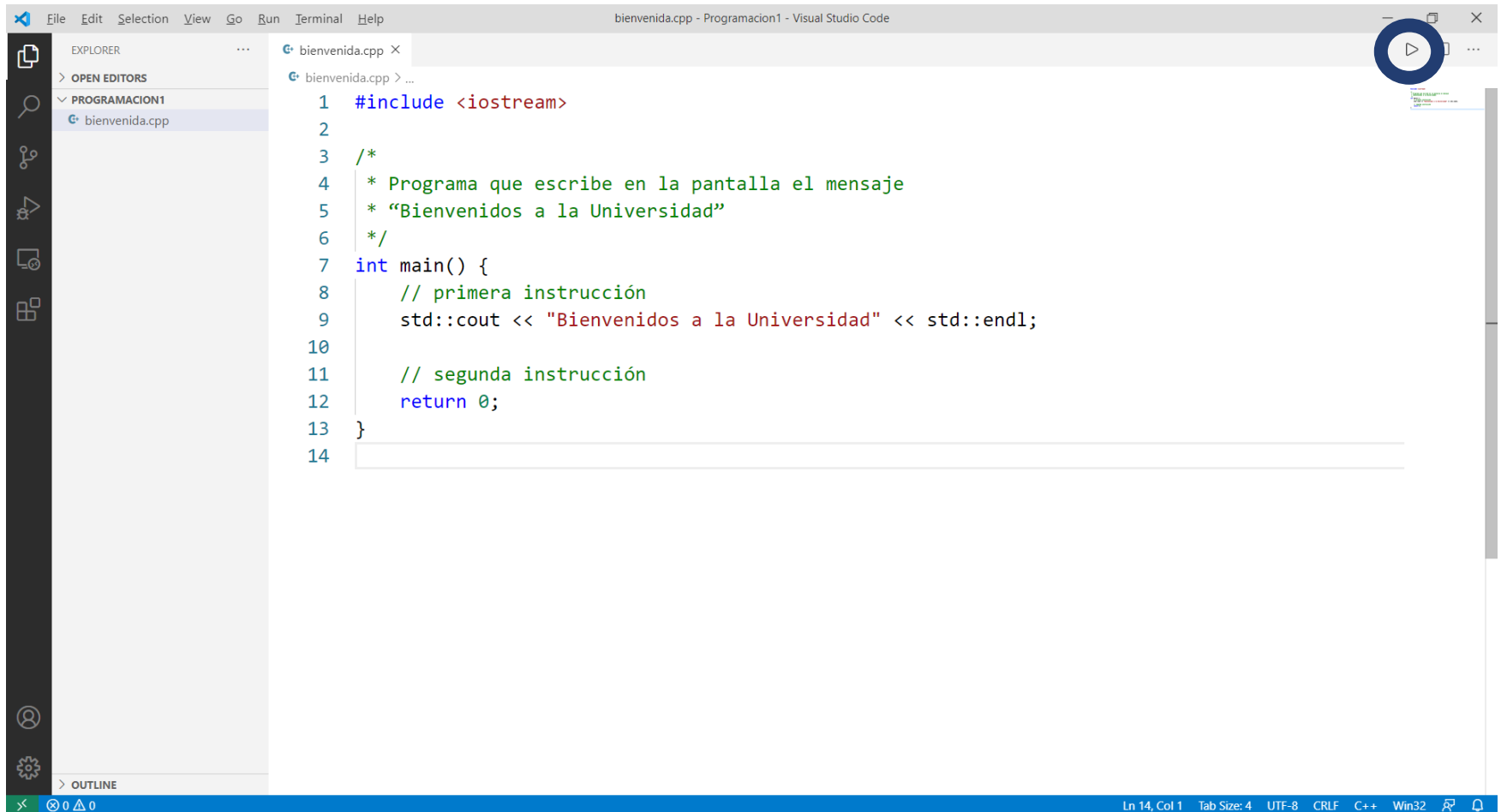
E:\Miguel\Documentos\Trabajo\Programacion1>bienvenida
Bienvenidos a la Universidad

E:\Miguel\Documentos\Trabajo\Programacion1>_
```





# Compilación y ejecución en Visual Studio Code



The screenshot displays the Visual Studio Code editor with the file 'bienvenida.cpp' open. The Explorer sidebar on the left shows the project structure with 'PROGRAMACION1' and 'bienvenida.cpp'. The main editor area contains the following C++ code:

```
1 #include <iostream>
2
3 /*
4  * Programa que escribe en la pantalla el mensaje
5  * "Bienvenidos a la Universidad"
6  */
7 int main() {
8     // primera instrucción
9     std::cout << "Bienvenidos a la Universidad" << std::endl;
10
11     // segunda instrucción
12     return 0;
13 }
14
```

The status bar at the bottom indicates the current position (Ln 14, Col 1), tab size (4), encoding (UTF-8), line endings (CRLF), and the active language (C++).



# Compilación y ejecución en Visual Studio Code

The screenshot displays the Visual Studio Code editor with a C++ file named `bienvenida.cpp` open. The code is as follows:

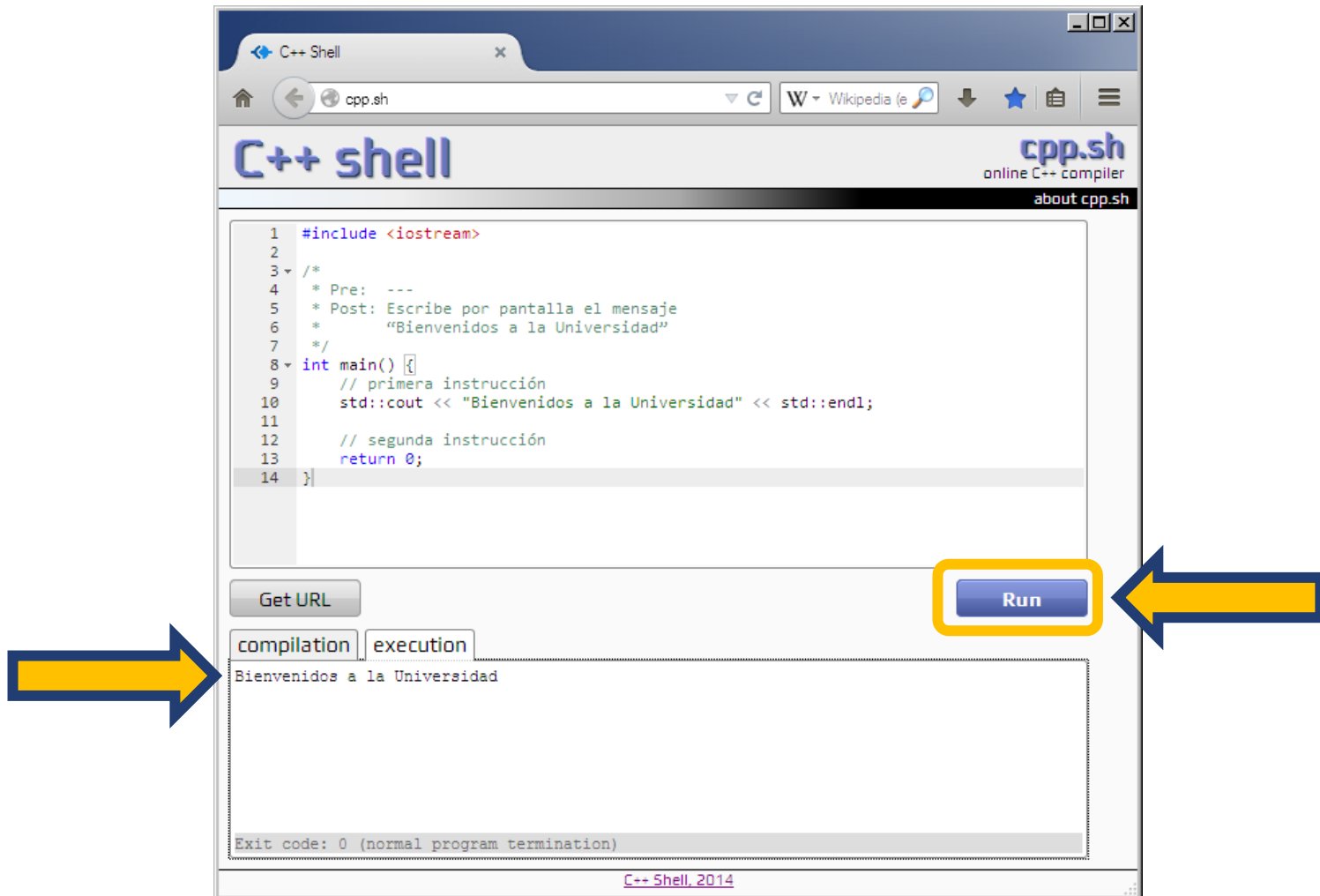
```
1 #include <iostream>
2
3 /*
4  * Programa que escribe en la pantalla el mensaje
5  * "Bienvenidos a la Universidad"
6  */
7 int main() {
8     // primera instrucción
9     std::cout << "Bienvenidos a la Universidad" << std::endl;
10
11     // segunda instrucción
12     return 0;
13 }
14
```

The Explorer sidebar on the left shows the project structure with `PROGRAMACION1` containing `bienvenida.cpp` and `bienvenida.exe`. The bottom panel shows the Windows PowerShell terminal with the following output:

```
PS E:\Miguel\Documentos\Trabajo\Programacion1> cd "e:\Miguel\Documentos\Trabajo\Programacion1" ; if ($?) { g++ bienvenida.cpp -o bienvenida } ; if ($?) { .\bienvenida }
Bienvenidos a la Universidad
PS E:\Miguel\Documentos\Trabajo\Programacion1>
```

A large yellow arrow points from the bottom left towards the terminal output.

# Compilación y ejecución en cpp.sh



The screenshot shows the cpp.sh online C++ compiler interface. The browser window has a single tab titled "C++ Shell" and the address bar shows "cpp.sh". The page header includes the "C++ shell" logo and the text "online C++ compiler" and "about cpp.sh". The main area contains a code editor with the following C++ code:

```
1 #include <iostream>
2
3 /*
4  * Pre: ---
5  * Post: Escribe por pantalla el mensaje
6  *       "Bienvenidos a la Universidad"
7  */
8 int main() {
9     // primera instrucción
10    std::cout << "Bienvenidos a la Universidad" << std::endl;
11
12    // segunda instrucción
13    return 0;
14 }
```

Below the code editor, there is a "Get URL" button and two tabs: "compilation" and "execution". The "execution" tab is selected, and the output area displays "Bienvenidos a la Universidad". At the bottom of the output area, it says "Exit code: 0 (normal program termination)". A yellow arrow points to the "Run" button, and another yellow arrow points to the output area.

# Un primer programa en C++

```
#include <iostream>

/*
 * Programa que escribe en la pantalla el mensaje
 * "Bienvenidos a La Universidad"
 */
int main() {
    // primera instrucción
    std::cout << "Bienvenidos a la Universidad" << std::endl;

    // segunda instrucción
    return 0;
}
```



# Un primer programa en C++

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

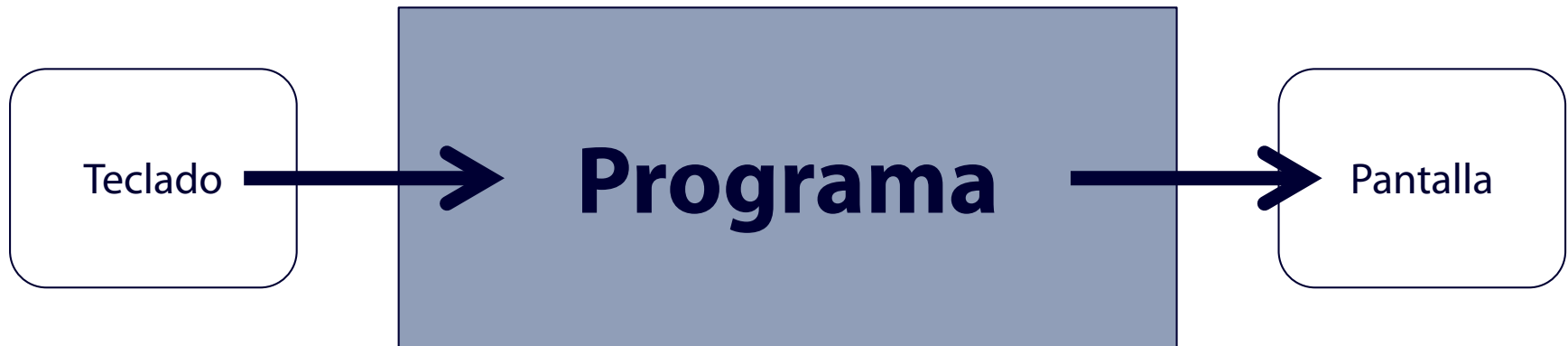
/*
 * Programa que escribe en la pantalla el mensaje
 * "Bienvenidos a la Universidad"
 */
int main() {
    // primera instrucción
    cout << "Bienvenidos a la Universidad" << endl;

    // segunda instrucción
    return 0;
}
```



# Un programa interactivo que lee del teclado

---



# Un programa interactivo que lee del teclado

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

/*
 * Programa que solicita un número entero al usuario y lo
 * escribe en pantalla.
 */
int main() {
    cout << "Escriba un número entero: ";

    int numero;
    cin >> numero;

    cout << "El número escrito es el " << numero << endl;

    return 0;
}
```



# Un programa interactivo que lee del teclado. Ejecución

Escriba un número entero:





# Un programa interactivo que lee del teclado. Ejecución

Escriba un número entero: 2019



# Un programa interactivo que lee del teclado. Ejecución

Escriba un número entero: 2019↵



# Un programa interactivo que lee del teclado. Ejecución

Escriba un número entero: 2019

El número escrito es el 2019

# Programa interactivo que hace un cálculo

```
#include <iostream>
using namespace std;
const double PI = 3.14159265358979323846;
/*
 * Solicita al usuario la longitud de un radio y escribe en la
 * pantalla el área del círculo correspondiente.
 */
int main() {
    cout << "Escriba el radio de un círculo: ";

    double r;
    cin >> r;

    cout << "El área de un círculo de radio " << r << " es "
         << PI * r * r << endl;

    return 0;
}
```



# Programa interactivo que hace un cálculo. Ejecución

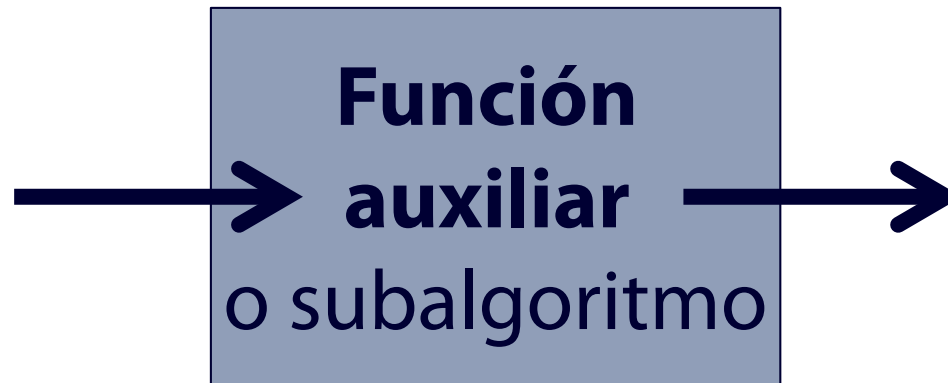
Escriba el radio de un círculo: 2.5

El área de un círculo de radio 2.5 es 19.635



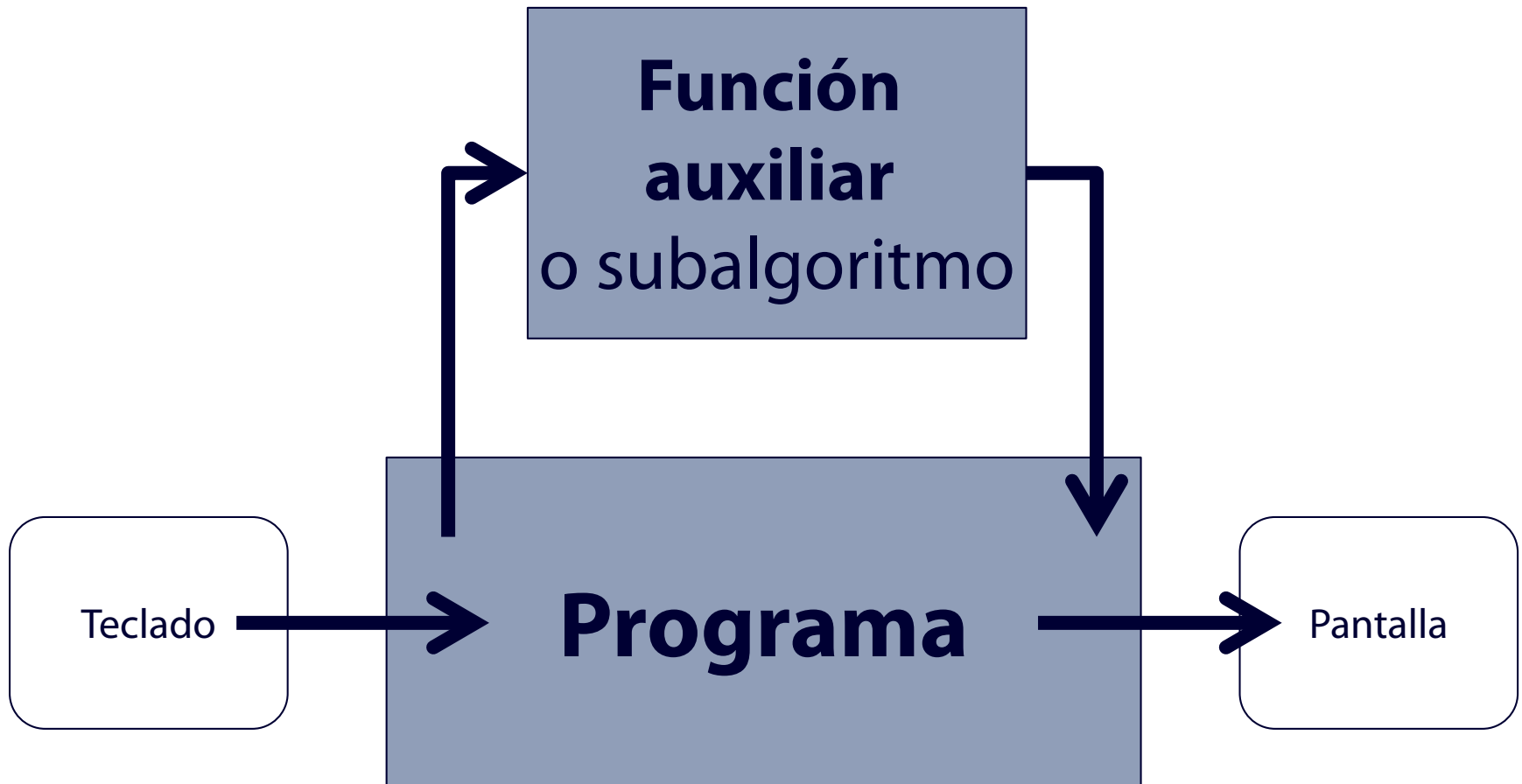
# Un programa que utiliza una función auxiliar

---





# Un programa que utiliza una función auxiliar



# Un programa que utiliza una función auxiliar

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

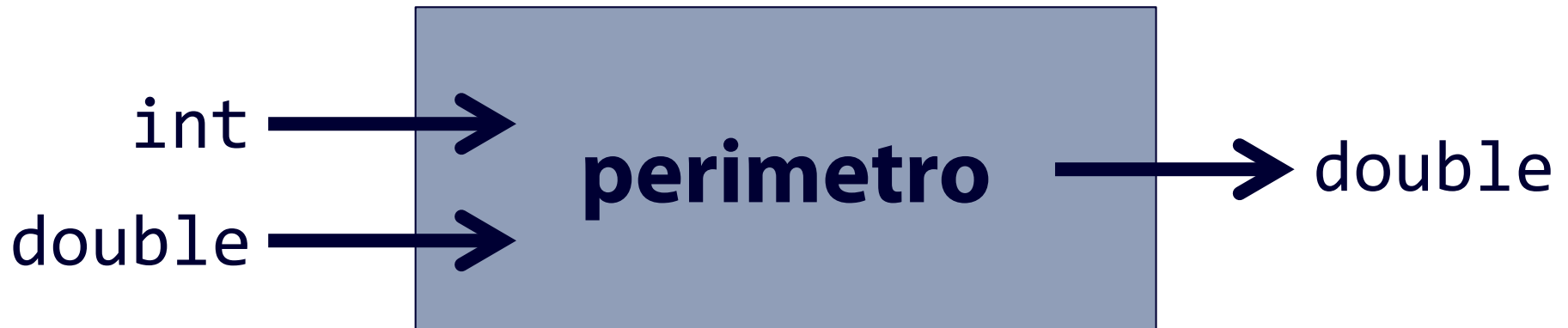
/*
 * Dado un polígono regular con un número de lados
 * igual al valor del parámetro «numLados» de
 * longitud igual al valor del parámetro «longitud»,
 * devuelve el perímetro de dicho polígono regular.
 * «numLados» tiene que ser mayor o igual que 3 y
 * «longitud» mayor que 0.0.
 */
double perimetro(int numLados, double longitud) {
    return numLados * longitud;
}
```





# Un programa que utiliza una función auxiliar

---



# Un programa que utiliza una función auxiliar

```
/*  
 * Programa que solicita al usuario el número de lados de un  
 * polígono regular y la longitud de sus lados y escribe en la  
 * pantalla el área de dicho polígono regular.  
 */  
int main() {  
    cout << "Escriba el número de lados: ";  
    int numLados;  
    cin >> numLados;  
  
    cout << "Escriba la longitud del lado: ";  
    double longitud;  
    cin >> longitud;  
  
    cout << "El perímetro de un polígono regular de " << numLados  
        << " de longitud " << longitud << " es "  
        << perimetro(numLados, longitud) << "." << endl;  
  
    return 0;  
}
```



# Un programa que utiliza una función auxiliar. Ejecución

Escriba el número de lados: 6

Escriba la longitud del lado: 2.5

El perímetro de un polígono regular de 6 lados de longitud 2.5 es 15.

# Funciones y procedimientos

---

- ❑ **Algoritmos** que resuelven un problema concreto de tratamiento de información
- ❑ Pueden invocarse unos a otros
- ❑ En C++, siempre los denominaremos funciones

# Funciones y procedimientos

---

- Trabajan con datos
  - Datos de **entrada**
    - Parámetros
    - Datos leídos de teclado
  - Datos de **salida**
    - Valor devuelto
    - Datos escritos en la pantalla
- Tienen un **cuerpo**
  - Acciones algorítmicas que manipulan los datos para resolver el problema concreto
- Se describen con una **especificación**
  - Documentación sobre lo que realiza el procedimiento o función

# Propiedades de un algoritmo

---

## □ **Imprescindibles**

- Corrección
- Legibilidad

## □ **Deseables**

- Generalidad
- Reusabilidad
- Eficiencia
- Independencia de la máquina y del lenguaje
- Simplicidad
- Fiabilidad

# Programa sintácticamente incorrecto

```
#include <iostream>

/*
 * Programa que escribe en la pantalla el mensaje
 * "Bienvenidos a La Universidad"
 */
{
    // primera instrucción
    cout << "Bienvenidos a La Universidad" << endl;

    // segunda instrucción
    return 0;
}
```

# Programa formalmente incorrecto

```
#include <iostream>

using namespace std;

/*
 * Programa que escribe en la pantalla la suma de los
 * números del 1 al 5.
 */
int main() {
    cout << 1 + 2 + 3 + 4 << endl;
    return 0;
}
```



# Propiedades de un algoritmo

---

## □ **Imprescindibles**

- Corrección
- **Legibilidad**

## □ **Deseables**

- Generalidad
- Reusabilidad
- Eficiencia
- Independencia de la máquina y del lenguaje
- Simplicidad
- Fiabilidad

# Programa C++ ilegible

```
#include <iostream>
#include <iomanip>
using namespace std; void o(double oo) {
const double ooo=3.14159265358979323846;
cout<<setw(7)<<oo<<setw(16)<<2.0*ooo*oo<<
endl;}int main(){cout<<setprecision(2); cout
<<fixed; cout<<setw(7)<<"Radi o"<<setw(20)<<
"Circunferenci a"<<endl; cout<<setw(7)<<
"====="<<setw(20)<<"===== " <<endl; o
(1.234); o(5.012); o(11.5178); cout<<endl;
return 0; }
```

# Propiedades de un algoritmo

---

## □ **Imprescindibles**

- Corrección
- Legibilidad

## □ **Deseables**

- **Generalidad**
- **Reusabilidad**
- Eficiencia
- Independencia de la máquina y del lenguaje
- Simplicidad
- Fiabilidad

# Generalidad

```
/*  
 * Pre: ---  
 * Post: Devuelve la suma de los enteros  
 * comprendidos en el intervalo [1, 100]  
 */  
int sumaDe1A100() {  
    int resultado = 0;  
    for (int i = 1; i <= 100; i++) {  
        resultado = resultado + i;  
    }  
    return resultado;  
}
```

# Generalidad

```
/*  
 * Pre: inicial <= final  
 * Post: Ha devuelto la suma de los enteros  
 * comprendidos en el intervalo [inicial, final].  
 */  
int suma(int inicial, int final) {  
    int resultado = 0;  
    for (int i = inicial; i <= final; i++) {  
        resultado = resultado + i;  
    }  
    return resultado;  
}
```

# Propiedades de un algoritmo

---

## □ **Imprescindibles**

- Corrección
- Legibilidad

## □ **Deseables**

- Generalidad
- Reusabilidad
- **Eficiencia**
- Independencia de la máquina y del lenguaje
- Simplicidad
- Fiabilidad

# Eficiencia

```
/*  
 * Pre: inicial <= final  
 * Post: Ha devuelto la suma de los enteros  
 * comprendidos en el intervalo  
 * [inicial, final].  
 */  
int sumaEficiente(int inicial, int final) {  
    return (inicial + final)  
        * (final - inicial + 1) / 2;  
}
```

# Propiedades de un algoritmo

---

## □ **Imprescindibles**

- Corrección
- Legibilidad

## □ **Deseables**

- Generalidad
- Reusabilidad
- Eficiencia
- **Independencia** de la máquina y del lenguaje
- **Simplicidad**
- **Fiabilidad**



# Resumen

---

- ❑ Problemas de tratamiento de información
- ❑ Algoritmos y programas
- ❑ Ejemplos de programas C++
- ❑ Propiedades de un algoritmo

# ¿Cómo se estudia este tema?

---

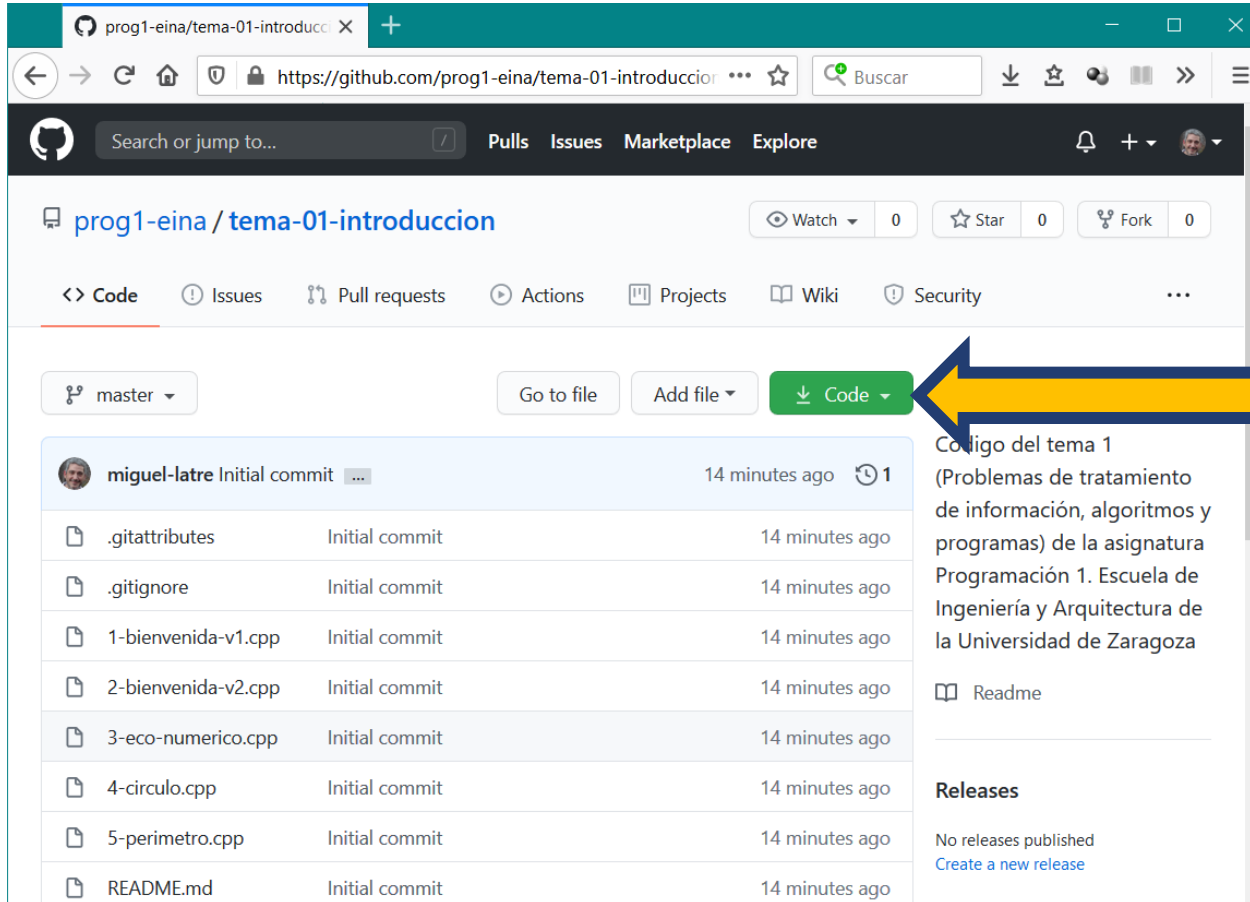
- Repasando las transparencias
- Leyendo las secciones 1.1, 1.2 y 1.4 del capítulo 1 de los apuntes del profesor Martínez, en Moodle
- Ejecutando los programas presentados en un entorno de ejecución en línea, como <http://cpp.sh/>
- Ejecutando los programas en Visual Studio Code:
  - Instalándolo según las instrucciones de «[Tutorial para la instalación de Visual Studio Code](#)» en Moodle
  - Leyendo la descripción del entorno Visual Studio Code de la 1.ª práctica de la asignatura.
    - Disponible el viernes en Moodle

# ¿Cómo se estudia este tema?

---

- Código fuente de este tema
  - Se puede copiar y pegar
  - También está disponible en el repositorio <https://github.com/prog1-eina/tema-01-introduccion>

# ¿Cómo se estudia este tema?



prog1-eina/tema-01-introduccion

Search or jump to... Pulls Issues Marketplace Explore

Watch 0 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security

master

Go to file Add file Code

miguel-latre Initial commit 14 minutes ago 1

File	Commit	Time
.gitattributes	Initial commit	14 minutes ago
.gitignore	Initial commit	14 minutes ago
1-bienvenida-v1.cpp	Initial commit	14 minutes ago
2-bienvenida-v2.cpp	Initial commit	14 minutes ago
3-eco-numerico.cpp	Initial commit	14 minutes ago
4-circulo.cpp	Initial commit	14 minutes ago
5-perimetro.cpp	Initial commit	14 minutes ago
README.md	Initial commit	14 minutes ago

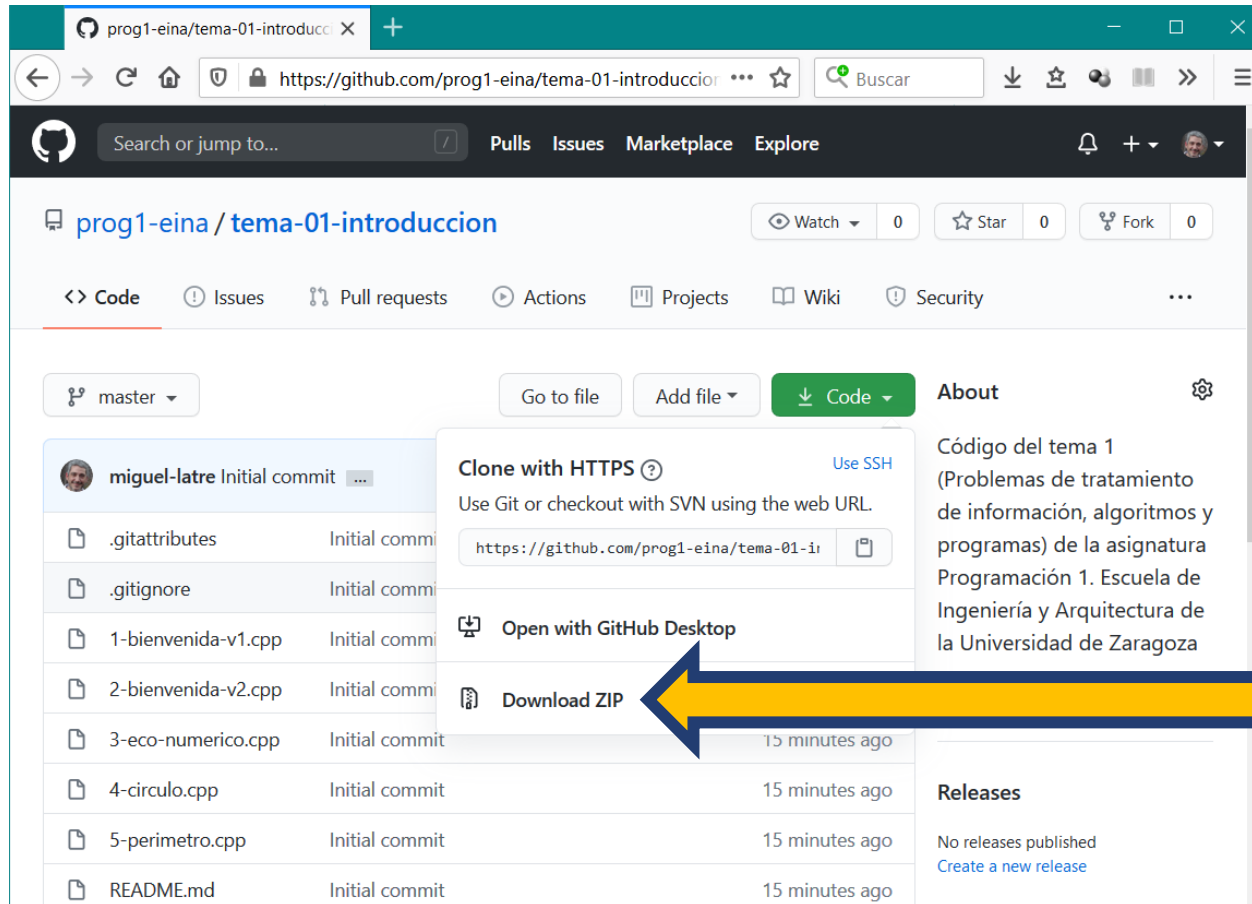
Codigo del tema 1  
(Problemas de tratamiento de información, algoritmos y programas) de la asignatura Programación 1. Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza

Readme

Releases

No releases published  
[Create a new release](#)

# ¿Cómo se estudia este tema?



The screenshot shows a web browser displaying the GitHub repository page for 'prog1-eina/tema-01-introduccion'. The repository is owned by 'miguel-latre' and contains several files, including '.gitattributes', '.gitignore', and several C++ source files. A 'Clone with HTTPS' dialog box is open, showing the repository URL and options to 'Open with GitHub Desktop' or 'Download ZIP'. A large yellow arrow points to the 'Download ZIP' button. The right side of the page contains an 'About' section with a description of the repository's content and a 'Releases' section.

prog1-eina/tema-01-introduccion

Search or jump to... Pulls Issues Marketplace Explore

Watch 0 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security

master

Go to file Add file Code

Clone with HTTPS ? Use SSH

Use Git or checkout with SVN using the web URL.

https://github.com/prog1-eina/tema-01-introduccion

Open with GitHub Desktop

Download ZIP

About

Código del tema 1  
(Problemas de tratamiento de información, algoritmos y programas) de la asignatura Programación 1. Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza

Releases

No releases published  
[Create a new release](#)

# ¿Cómo se estudia este tema?

---

- ❑ Descomprimir
- ❑ Abrir Visual Studio Code
- ❑ Menú «File» > «Open folder...»
- ❑ Buscar y seleccionar el directorio en el que se ha descomprimido