

# Clase 3: Algoritmia, lógica, pseudocódigo

Cómo piensan los computadores

# Qué cosas se le pueden ordenar a un computador

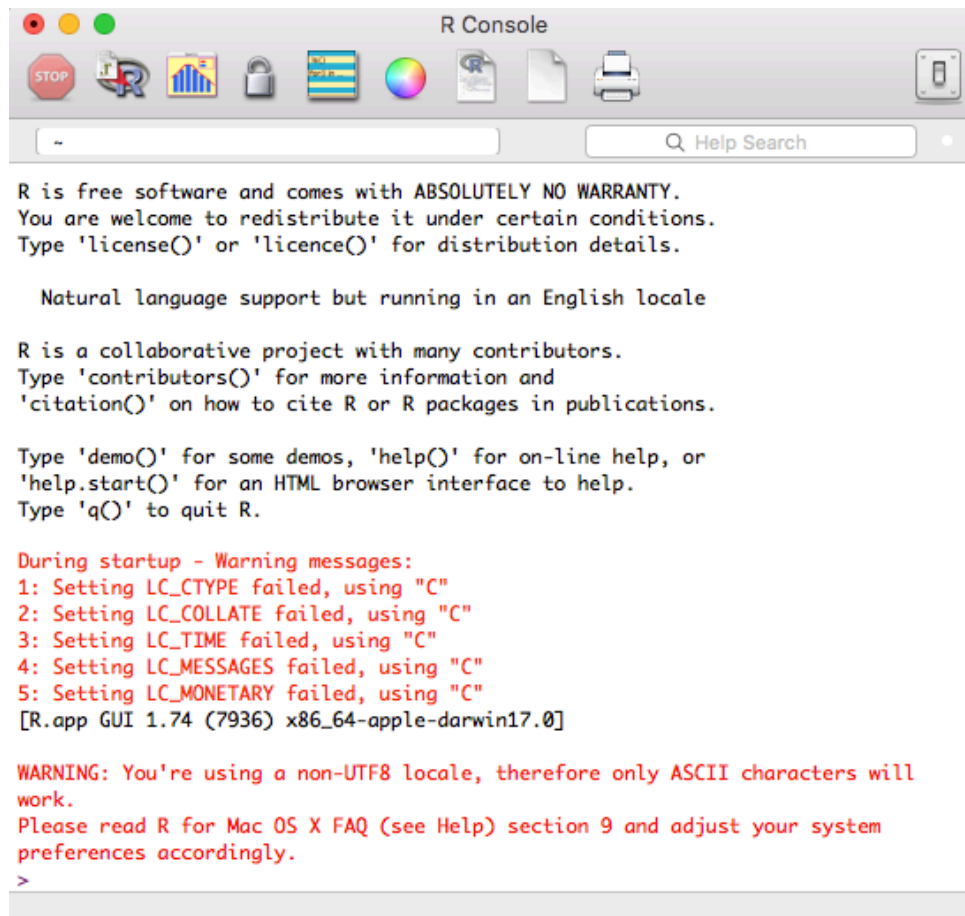
Ejecutar procesos computacionales como cálculos

Visualizar información (que puede ser resultado de los procesos): generación de gráficos

Dos cosas diferentes son:

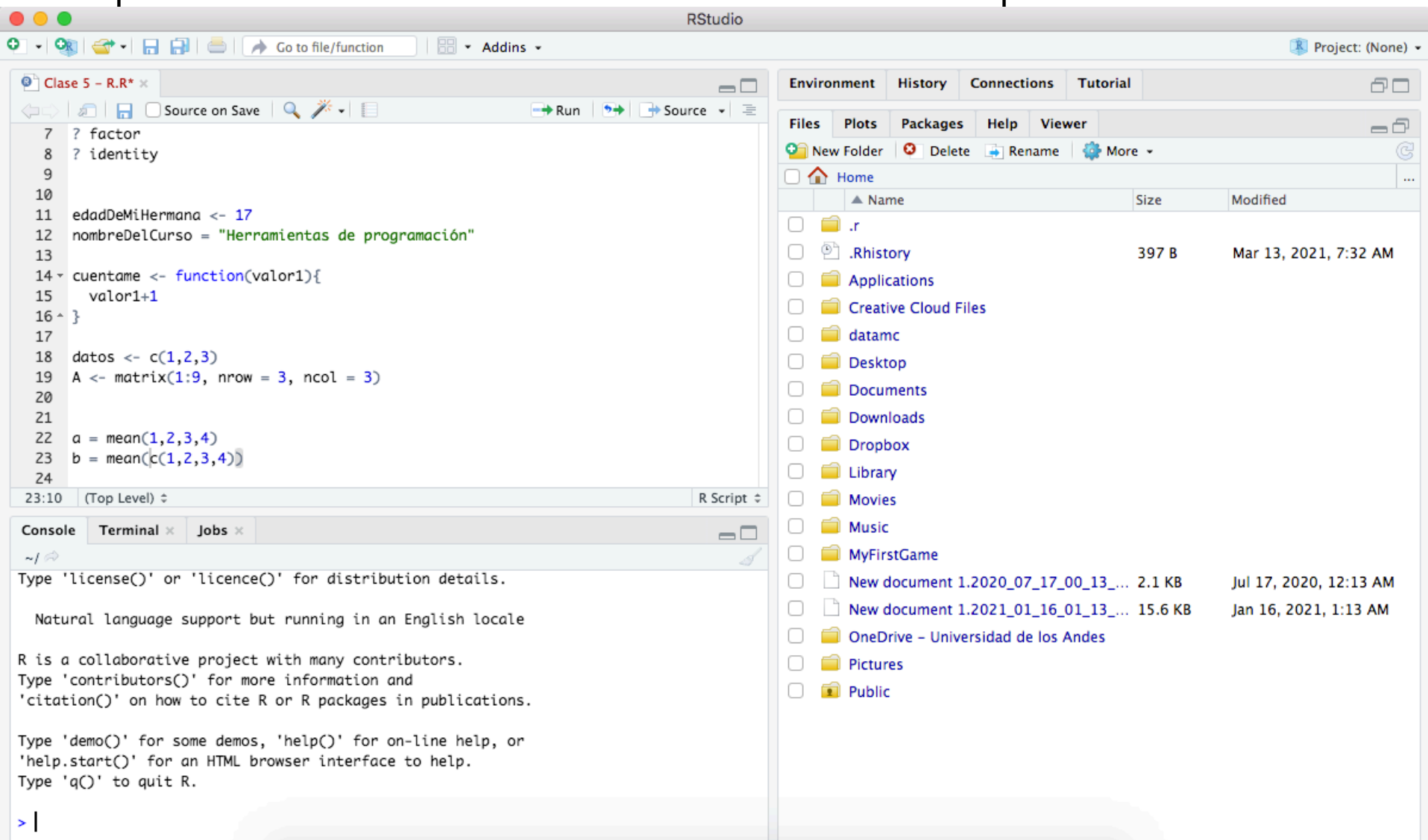
- Decirle al computador que **calcule**  $2+4$
- Decirle al computador que **muestre el valor** en pantalla

# Interfaz Gráfica: la(s) ventana(s) que permiten observar información del computador.

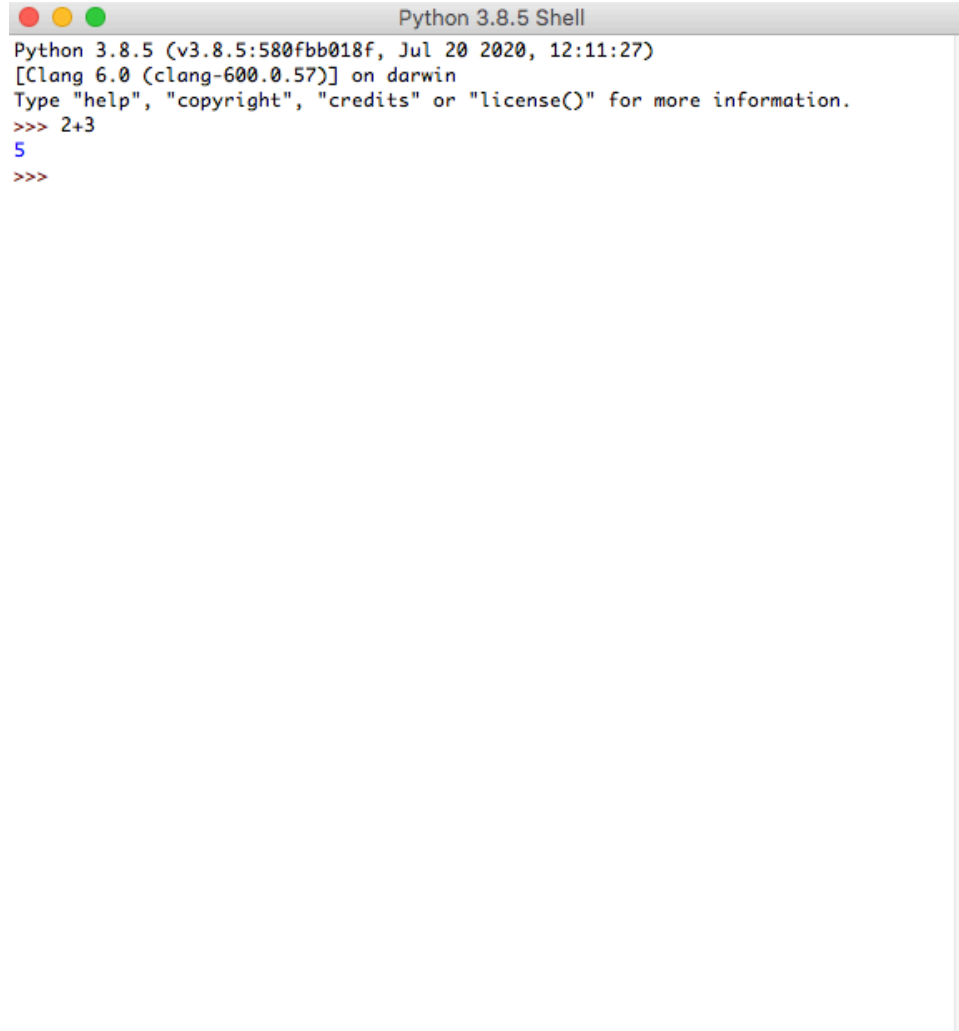


```
R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.  
  
Natural language support but running in an English locale  
  
R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.  
  
Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.  
  
During startup - Warning messages:  
1: Setting LC_CTYPE failed, using "C"  
2: Setting LC_COLLATE failed, using "C"  
3: Setting LC_TIME failed, using "C"  
4: Setting LC_MESSAGES failed, using "C"  
5: Setting LC_MONETARY failed, using "C"  
[R.app GUI 1.74 (7936) x86_64-apple-darwin17.0]  
  
WARNING: You're using a non-UTF8 locale, therefore only ASCII characters will  
work.  
Please read R for Mac OS X FAQ (see Help) section 9 and adjust your system  
preferences accordingly.  
>
```

# Interfaz Gráfica: la(s) ventana(s) que permiten observar información del computador.

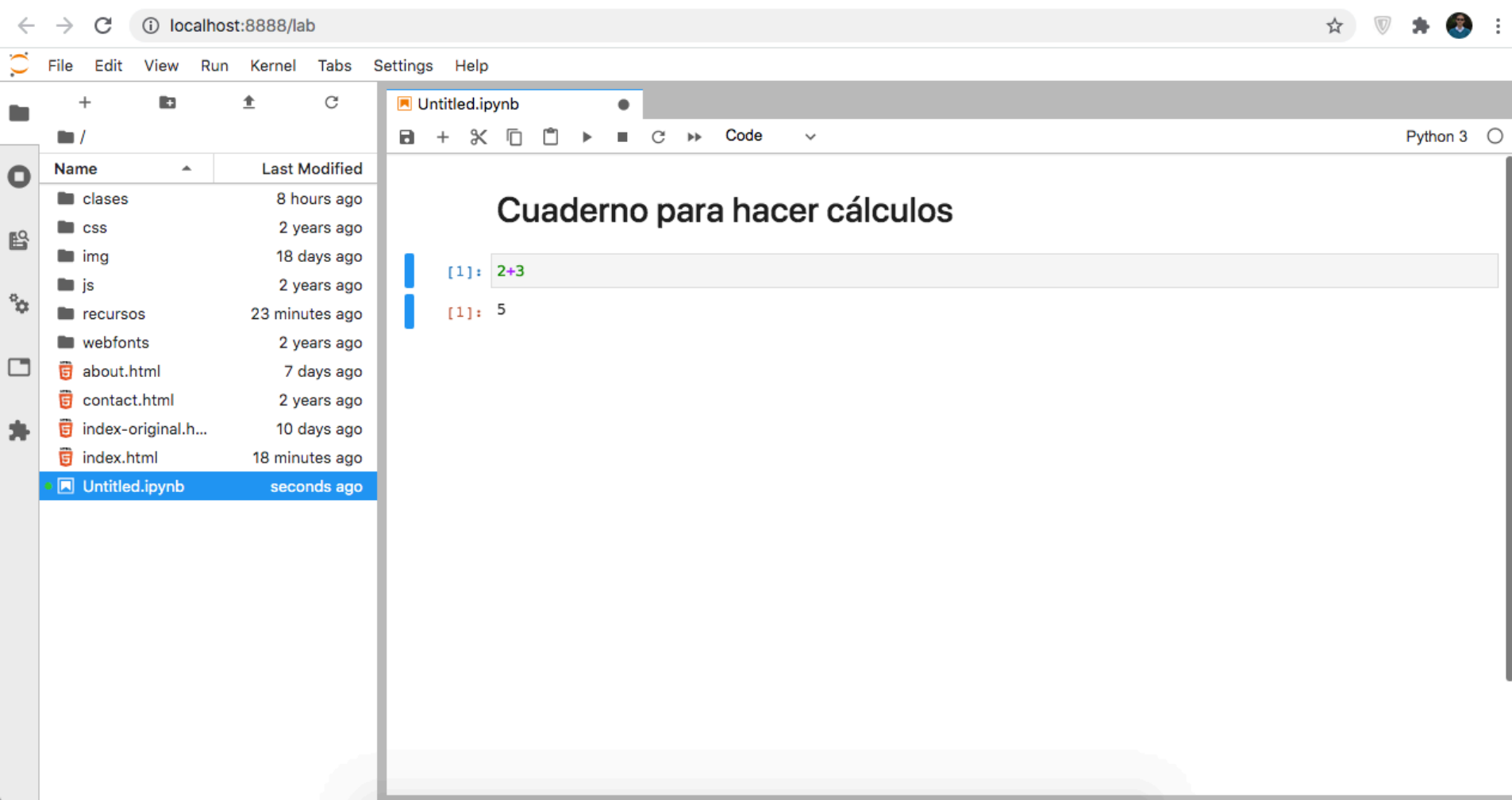


# Interfaz Gráfica: la(s) ventana(s) que permiten observar información del computador.

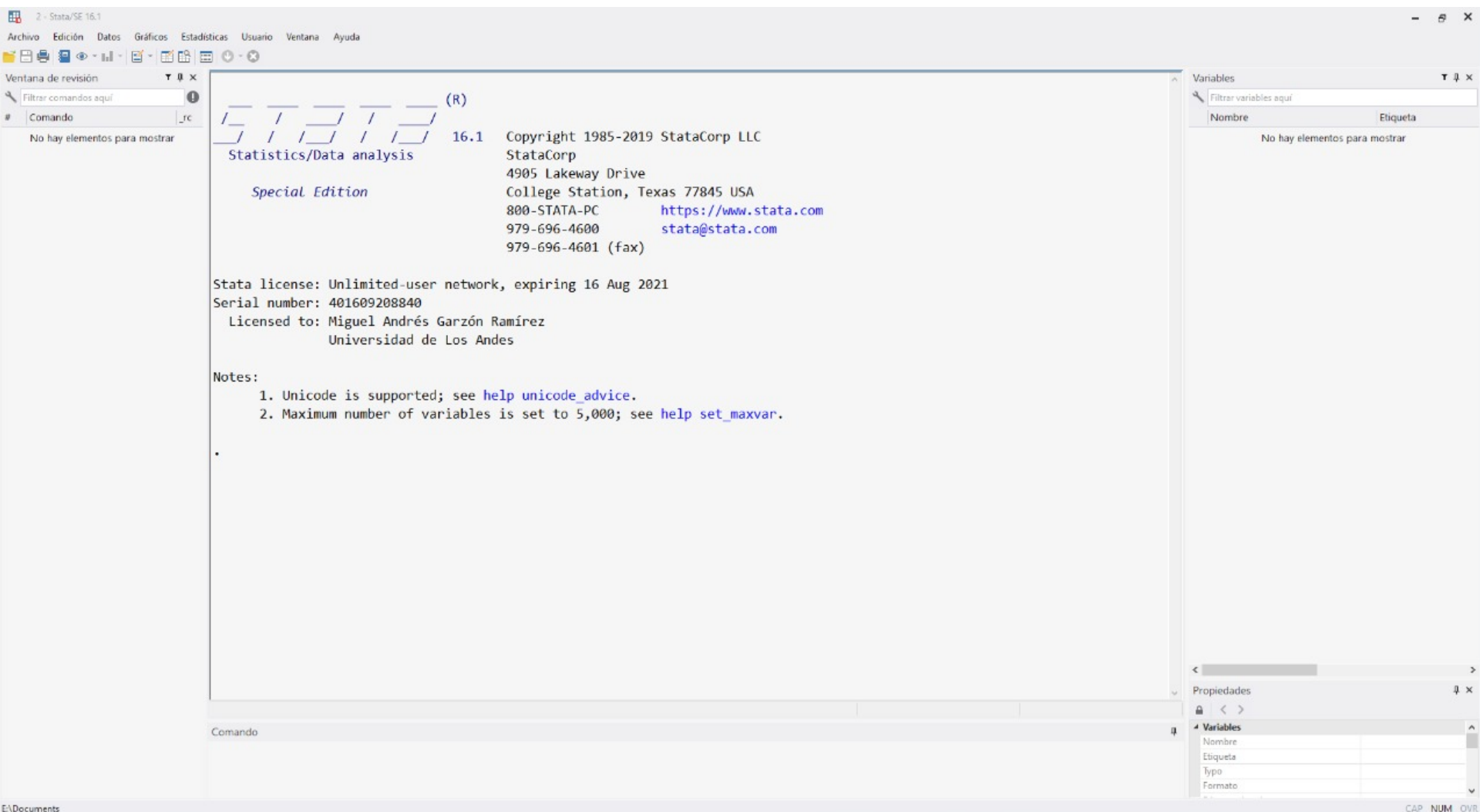
A screenshot of a macOS-style window titled "Python 3.8.5 Shell". The window has a title bar with red, yellow, and green window control buttons. The content area shows the following text:

```
Python 3.8.5 (v3.8.5:580fbb018f, Jul 20 2020, 12:11:27)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> 2+3
5
>>>
```

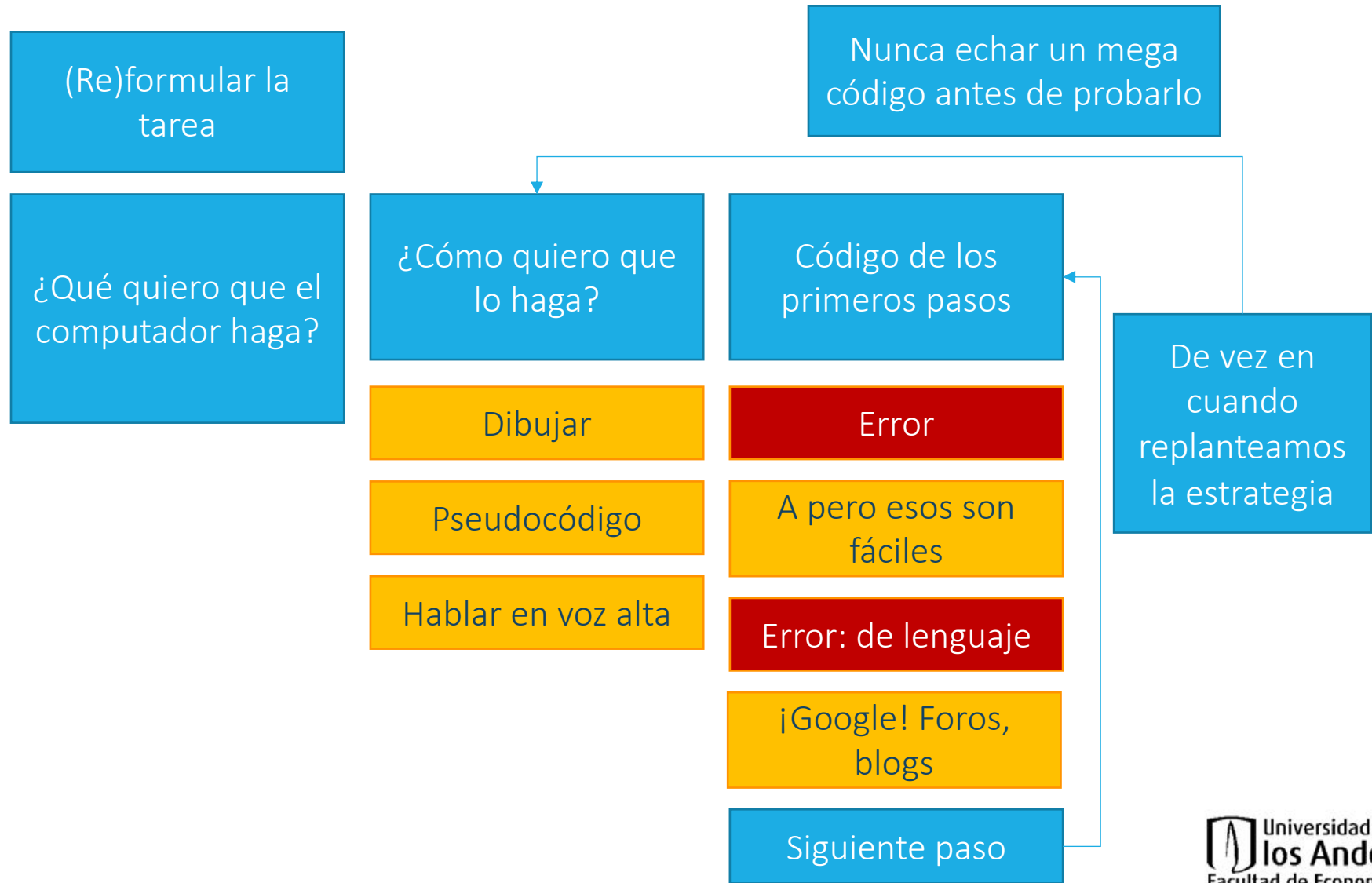
# Interfaz Gráfica: la(s) ventana(s) que permiten observar información del computador.



# Interfaz Gráfica: la(s) ventana(s) que permiten observar información del computador.



# Primero: cómo es programar





# Cómo se le dan instrucciones al computador

- Instrucciones se ejecutan **en un orden**
- Se puede **consultar y almacenar** información
- Se pueden construir **múltiples caminos**
- Se pueden repetir las **mismas instrucciones** cuando sean útiles

La vez pasada vimos que un **algoritmo**: una secuencia de instrucciones paso a paso para ejecutar una tarea.

# Vamos a ver de primera mano qué es un algoritmo



# Algoritmos

Noten que **le dimos instrucciones** al gato

Noten que **repetimos** instrucciones a veces

Noten que **preguntamos** cosas a veces

# Vamos a registrarnos en Scratch

En la página del curso encuentran las instrucciones para registrarse en <https://scratch.mit.edu/>

Tenemos 7 minutos para esto.

# Cómo piensa un computador

Cada lenguaje tendrá una forma de “escribir” algoritmos con estos elementos

# Para calcular hay que recordar

Hay dos tipos de información con la que van a trabajar:

| Columna 1 | Columna 2 | Columna 3 |
|-----------|-----------|-----------|
| 1         | 234.23    | "papas"   |
| 3         | 534.55    | "tajadas" |
| 4         | 323.86    | "papas"   |

DATOS

numero-de-repeticiones = 23  
datos-viejos = datosCargados  
datos-nuevos = datosDeCarlos  
precio-del-dolar = 3532  
vamos-en = "Las estadísticas descriptivas"

VARIABLES

En ocasiones, van a querer dejar información inmutable (para que no cambie en toda la ejecución):

**constantes**

# No todo dato es del mismo tipo

¡Porque se usan de manera distinta!

- Con un número (digamos 2) puedo hacer cosas como:

$$\text{número} + 3 = 5$$

$$\text{número} / 2 = 1$$

número pero con dos decimales en lugar de tres

- Con un “texto” como: Si uno intenta manipular un dato inapropiadamente, el programa le va a mostrar un elegante y sofisticado error

# Los tipos de dato más comunes

- **Integer (int)**, entero  
2
- **Coma flotante (float)**, es un tipo de número decimal  
4,45234523
- **Character (char)**, es un caracter solito  
a
- **Cadena de caracteres (String)**, es un texto  
“Este es un texto que puede incluir números como 43”
- **Valor lógico (boolean)**, representa sí/no, verdadero/falso  
False



¿Cuál es el resultado de?

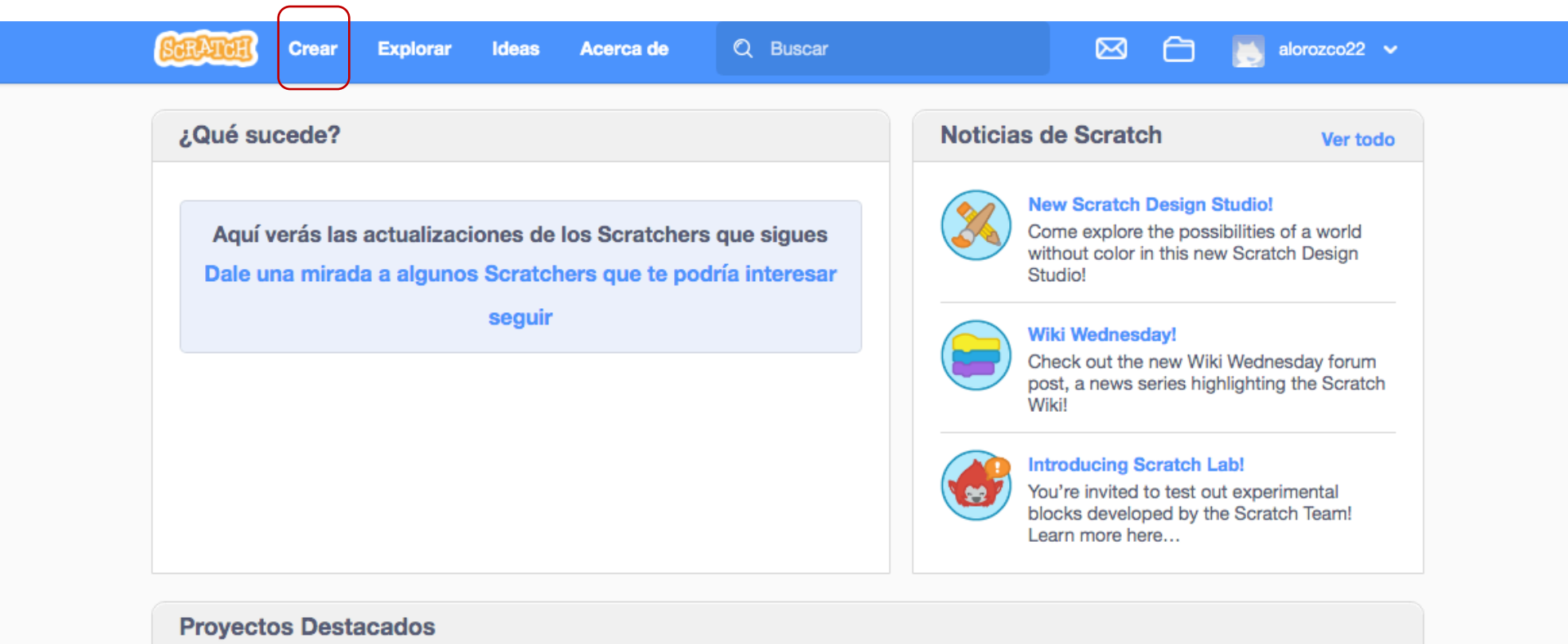
La operación “14” + “23”

Por favor anótenlo por ahí, tienen 10 segundos:

Respuesta:

“1423”

# Ejercicio con variables Scratch






**Scratch** **Crear** Explorar Ideas Acerca de Buscar

¿Qué sucede?

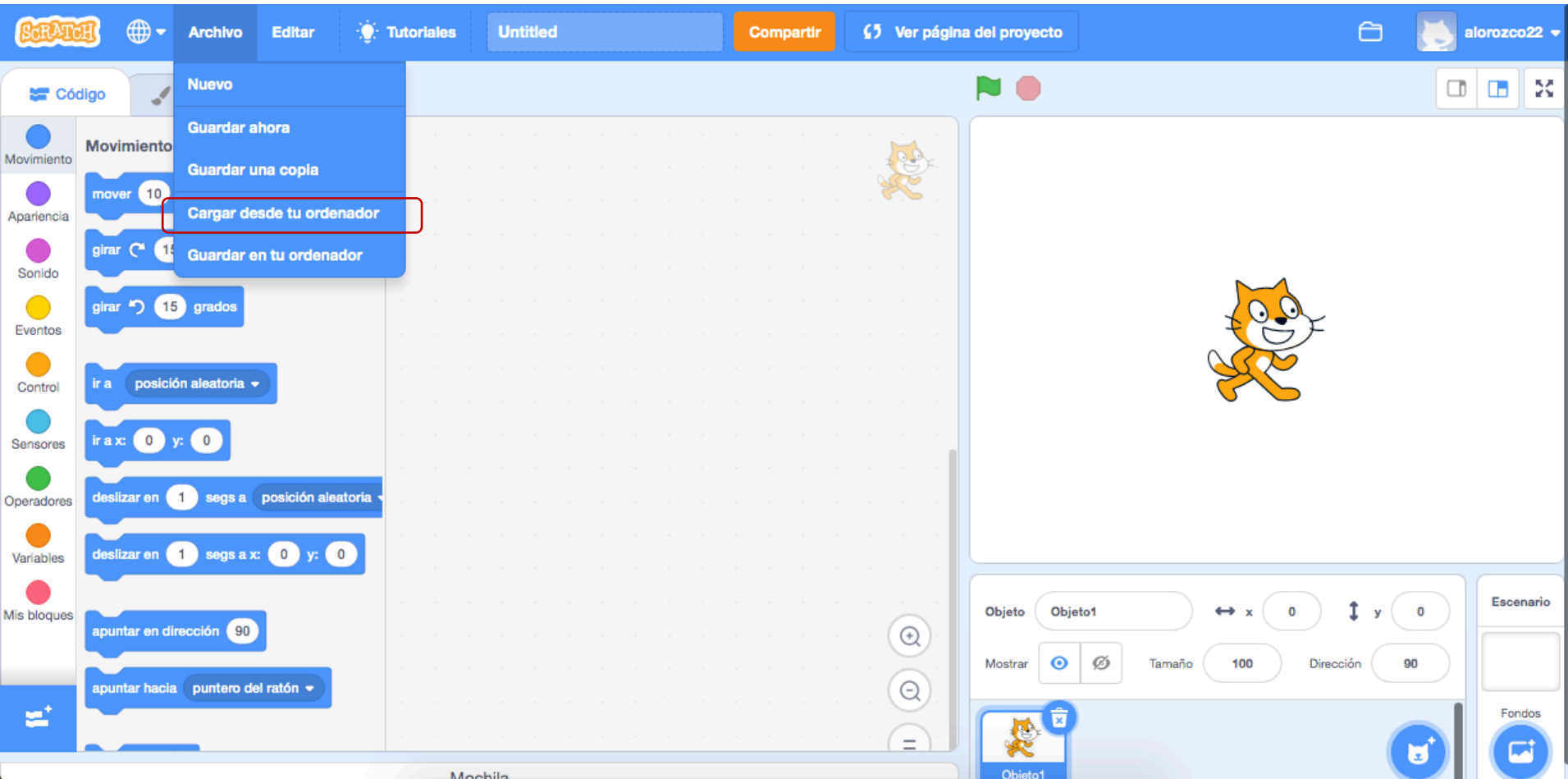
Aquí verás las actualizaciones de los Scratchers que sigues  
Dale una mirada a algunos Scratchers que te podría interesar seguir

**Noticias de Scratch** [Ver todo](#)

-  **New Scratch Design Studio!**  
Come explore the possibilities of a world without color in this new Scratch Design Studio!
-  **Wiki Wednesday!**  
Check out the new Wiki Wednesday forum post, a news series highlighting the Scratch Wiki!
-  **Introducing Scratch Lab!**  
You're invited to test out experimental blocks developed by the Scratch Team! Learn more here...

Proyectos Destacados

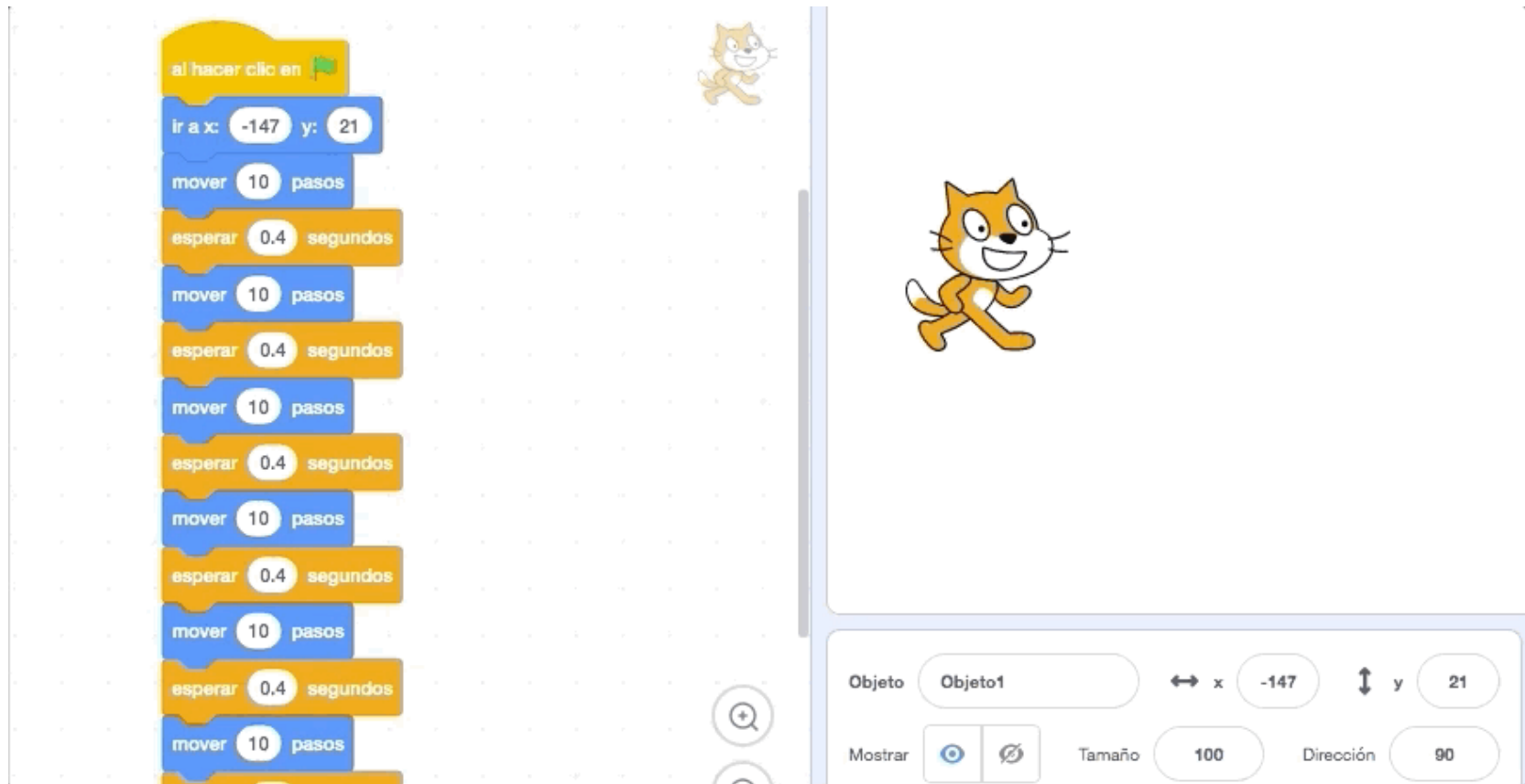
# Ejercicio con variables Scratch



# Repeticiones (loops)

- Cuando una instrucción es útil en más de una ejecución, hay dos formas de escribirlo

# Repeticiones (loops)



The image displays a Scratch project interface. On the left, a script is built on the 'al hacer clic en' (when clicked) event. The script consists of a sequence of blocks: 'Ir a x: -147 y: 21' (Go to x: -147 y: 21), followed by a loop of 'mover 10 pasos' (move 10 steps) and 'esperar 0.4 segundos' (wait 0.4 seconds) blocks. This sequence is repeated five times. On the right, the stage shows a cat sprite at the coordinates (-147, 21). The bottom right panel shows the object's properties: 'Objeto' (Object) is 'Objeto1', 'Mostrar' (Show) is checked, 'Tamaño' (Size) is 100, and 'Dirección' (Direction) is 90.

al hacer clic en

Ir a x: -147 y: 21

mover 10 pasos

esperar 0.4 segundos

mover 10 pasos

esperar 0.4 segundos

mover 10 pasos

esperar 0.4 segundos

mover 10 pasos

esperar 0.4 segundos

mover 10 pasos

Objeto Objeto1

Mostrar ☒ ☐

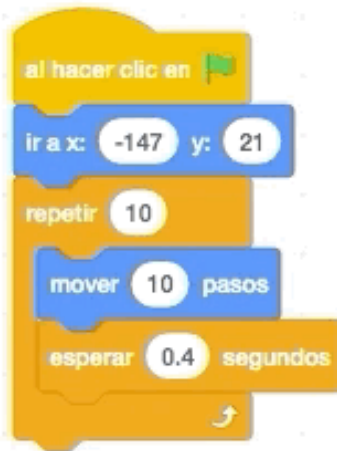
Tamaño 100

Dirección 90

x -147 y 21

# Repeticiones (loops)

- Cuando una instrucción es útil en más de una ejecución, hay dos formas de escribirlo

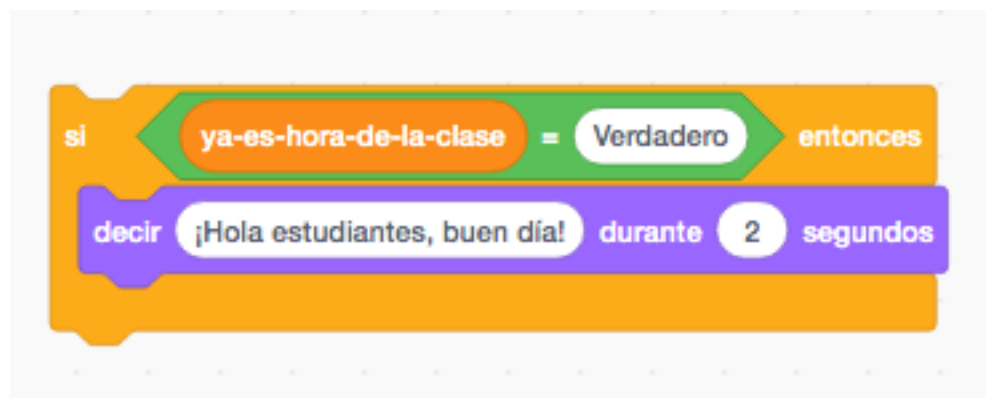


# Condicionales

- A veces vamos a necesitar evaluar las condiciones para poder tomar una decisión sobre qué hacer. En ese caso usamos condicionales.

# Condicionales: tipo if

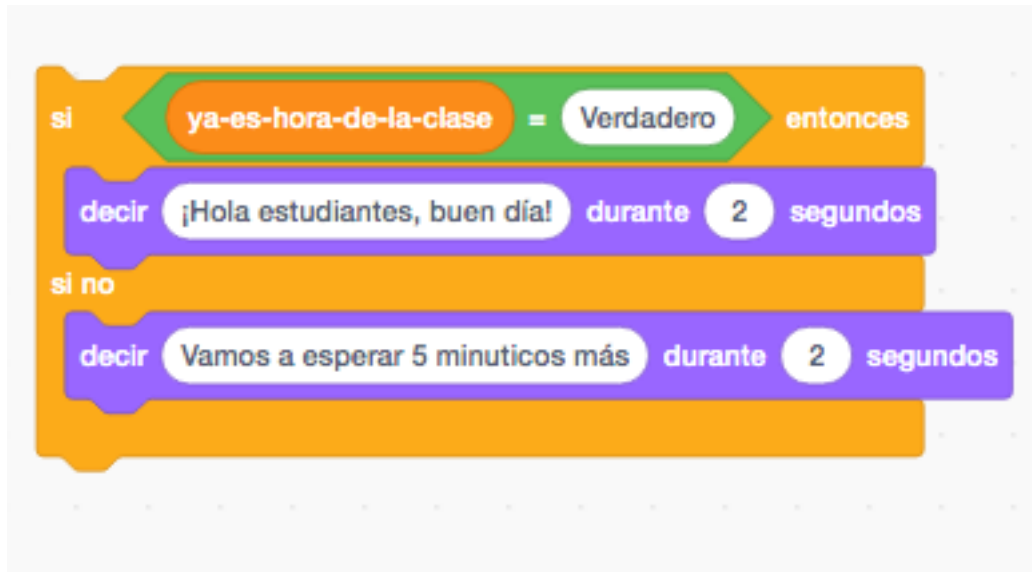
- A veces vamos a necesitar evaluar las condiciones para poder tomar una decisión sobre qué hacer. En ese caso usamos condicionales.





# Condicionales: tipo if-else

- A veces vamos a necesitar evaluar las condiciones para poder tomar una decisión sobre qué hacer. En ese caso usamos condicionales.



# Ejercicio 3



# Funciones / métodos

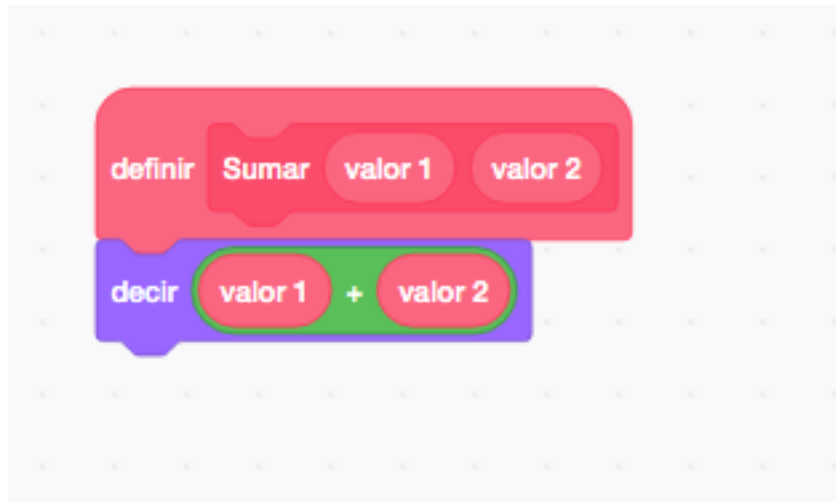
A veces va a ser útil tomar variables y usarlas para algo.

A veces, podemos devolverle al que nos preguntó el resultado de lo que hice con esos valores

# Funciones / métodos

A veces va a ser útil tomar variables y usarlas para algo.

A veces, podemos devolverle al que nos preguntó el resultado de lo que hice con esos valores

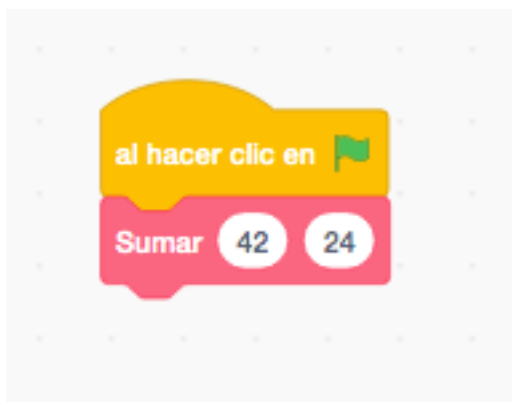


Primero las definimos

# Funciones / métodos

A veces va a ser útil tomar variables y usarlas para algo.

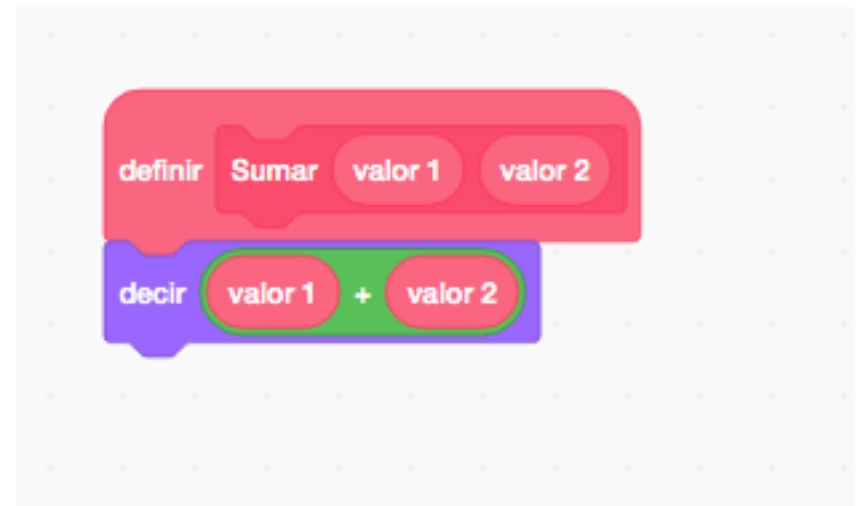
A veces, podemos devolverle al que nos preguntó el resultado de lo que hice con esos valores



Luego las usamos

# Scope local vs global

- Note que el valor1 o el valor2 no se pueden usar por fuera de la función. Porque no están definidos.
- Igual ocurría cuando definíamos una variable para este objeto vs para todos los objetos.



# Grupos de información

Listas, diccionarios, arreglos, matrices

# Cuando tenemos muchas variables

Si yo tengo algo del estilo:

Variable 1

Variable 2

Variable 3

Variable 4

Variable 5

Variable 6

Variable 7

Qué cansón manejar 7 variables que tienen la misma naturaleza



# Listas o arreglos: filas de valores

¡Ojo! Algunos lenguajes comienzan en 1, y otros en 0 a contar

Podemos crear listas del estilo:

Variables = [variable1, variable2, variable 3, ...]

¿Cuál es la ventaja?

Pásame por favor de la lista de variables, la que está en la posición 3

En una lista que cuenta desde  
cero

[1, 4, 5, 2, 6, 7]

¿Cuál es la longitud de la lista?

¿En qué posición está el último  
elemento? (el número 7)

# También hay estructuras como diccionarios

```
Juan = {dirección: "carrera 5",  
        edad: 32,  
        hermanos: False  
}
```

# Matrices

- Funcionan por filas y columnas:

$$\textit{Valores} = \begin{bmatrix} 23 & 12 \\ 57 & 42 \end{bmatrix}$$

¿El valor en la fila 1 y en la columna 2?

# Datos, cómo los representamos

- Si bien uno podría coger una tabla y ponerla en una matriz, cada lenguaje carga las bases de datos a su manera:
  - En R se parece a un objeto donde cada columna es un atributo **DataFrames**
  - En Stata se indica la **base de datos** y se juega con las columnas/variables
  - En Python también tiene **DataFrames**

# Operaciones para seleccionar datos

- Todo lenguaje (al menos aquí) permite seleccionar una sección de los datos. Hablar sobre esto.
- En Stata jugamos con **comandos**:
  - Hacer esto para aquellos datos que cumplen tal condición
- En python y R rebanamos primero la base:
  - Tome los datos **entre tanto y tanto**
  - Ejecute el cálculo con esos

Eso se conoce como indexación

# Algunos algoritmos son más rápidos que otros

- Hay muchas formas de lograr lo mismo
- **Ordenamiento**

<https://www.youtube.com/watch?v=lyZQPjUT5B4>

Vs

<https://www.youtube.com/watch?v=ywWBy6J5gz8>