Clase 02 - Algoritmos

IIC1103-07 - Introducción a la Programación

Cristian Ruz – cruz@ing.puc.cl

Jueves 8-Agosto-2019

Departamento de Ciencia de la Computación
Pontificia Universidad Católica de Chile

Contenidos

Contenidos

Contacto

- `iic1103@uc.cl`. Nicolás Quiroz
  - Coordinación
  - Justificación de inasistencias
  - Asistencia mal contada
  - Notas incorrectas en planilla
  - Recorrecciones
- `cruz@ing.puc.cl`. Profesor.
  - Materia
  - Situaciones especiales
- `rsalvarez@uc.cl`. Raúl Álvarez. Ayudante
  - Materia
  - Notas de participación
  - a.k.a. `KnowYourselves`

Contenidos

Nota de Participación

¿Recuerdan como pasar el curso?

$$\mathbf{NF} = 0.15 \cdot \mathbf{I1} + 0.15 \cdot \mathbf{I2} + 0.3 \cdot \mathbf{EX} + 0.1 \cdot \mathbf{T1} + 0.1 \cdot \mathbf{T2} + 0.1 \cdot \mathbf{T3} + 0.1 \cdot \mathbf{NP}$$

$$\mathbf{NF} \geq 4.0$$

$$\mathbf{Y}$$

$$(0.15 \cdot \mathbf{I1} + 0.15 \cdot \mathbf{I2} + 0.3 \cdot \mathbf{EX})/0.6 \geq 4.0$$

$$\mathbf{Y}$$

$$(\mathbf{T1} + \mathbf{T2} + \mathbf{T3})/3 \geq 4.0$$

No, el examen no es reprobatorio

Toda nota (parcial o final) será redondeada a 1 decimal

Nota de Participación (NP)

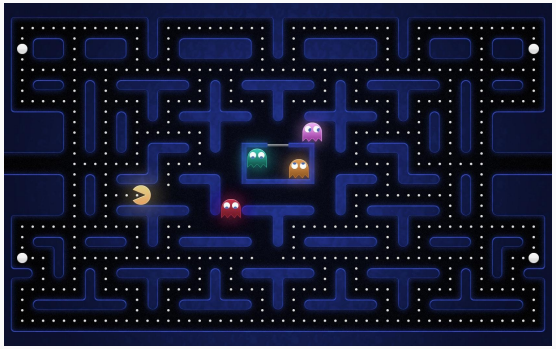
Se calculará en bases a **puntos de participación**.

- 1200 puntos máximo por resolver labs *hackerrank* antes de la fecha que será indicada.
- $\sim$100 puntos por respuesta a cuestionarios o actividades en clase (serán avisados)
- $\sim$400 puntos por desafíos en clases (serán anunciados)
- $\mathbf{NP} = \frac{\textbf{obtenidos}}{0.8 \times \textbf{puntos totales}} \times 6 + 1$

Puntos totales se definirá al final del semestre

Contenidos

Aprender a programar



Paso 1. Especificando una serie de pasos: un **algoritmo**

Algoritmo

- Secuencia ordenada de instrucciones
- ¿Cómo preparar un pie de limón?
    1. Preparar la masa
    2. Preparar relleno
    3. Preparar merengue
    4. Ponerlo en el horno
    5. Esperar

¿Cómo programar?

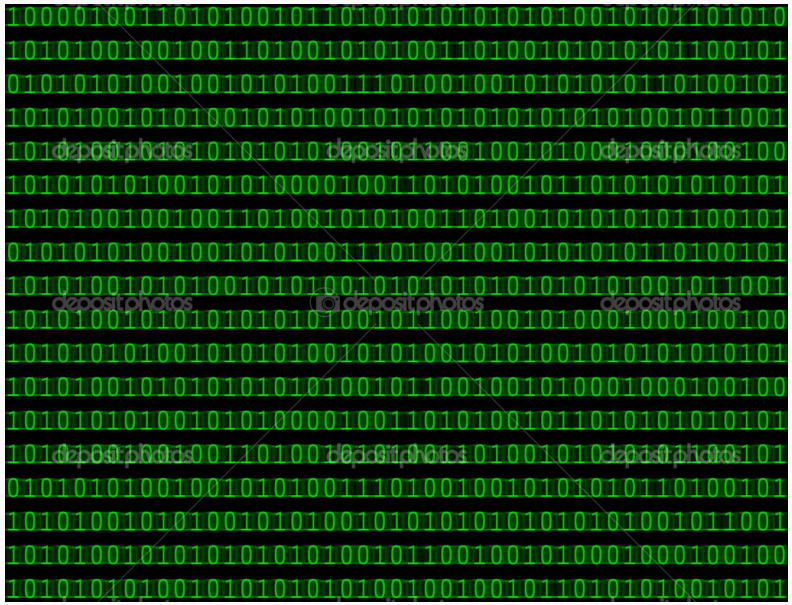
Paso 2. Usando una herramienta apropiada: Un computador

Un computador puede seguir instrucciones

Muy sencillo: decirle qué queremos hacer.

Especificándole un algoritmo

¿Cómo le decimos qué queremos hacer?
Hablando en su lenguaje: Lenguaje binario



¿Cómo programar?

En general no escribimos programas de esta manera.

- Código binario es un código de **bajo nivel**
- Humano usamos lenguaje natural: **lenguaje de alto nivel**
- Lamentablemente los computadores aún no son capaces de entender lenguaje natural (aún).

¿Cómo programar?

Pero podemos llegar a un punto intermedio:

Lenguajes de programación
Comprensibles por el computador, y por el humano

```
while True:
    print("Enter 'x' for exit.")
    check = input("Are you a Robot ? ")
    if check == 'x':
            break
    elif(check == 'yes'):
            print("Sorry!, you can not proceed.\n")
    else:
            print("Congrats!, you can proceed.\n")
```

Nuestro compromiso será aprender a hablarle al computador en un lenguaje de **alto nivel**.

Los algoritmos especificados en este lenguaje de programación se llamarán **programas**.

¿Cómo programar?

1. **Paso 1**. Especificar un **algoritmo**
2. **Paso 2**. Escribir el algoritmo en un **lenguage de programación**.
3. **Paso 3**. Ejecutar programa en un **computador**.

Lo que obtenemos como resultados es un **programa** que podemos **ejecutar** en un **computador**

¿Cómo programar?

¿En qué lenguaje?

¿Cómo programar?

¿En qué lenguaje?

¿Python?

¿Por qué python?

C++:

```cpp
#include <iostream>
int main() {
    std::cout << "Hello World!" << std::endl;
    std::cin.get();
    return 0;
}
```

¿Python?

¿Por qué python?

Java:

```java
public class HelloWorld {
  public static void main(String[] args) {
    System.out.println("Hello World!");
  }
}
```

¿Python?

¿Por qué python?

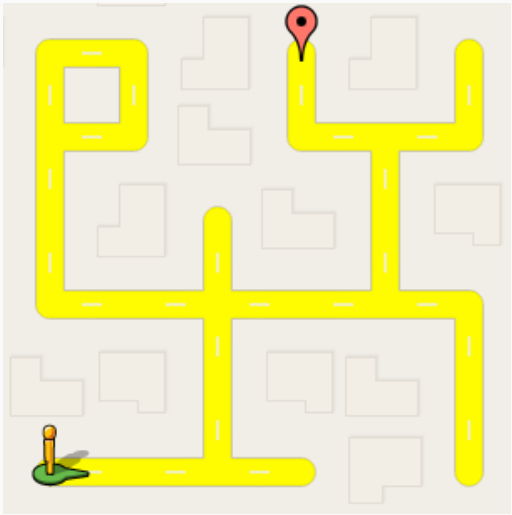
Python:

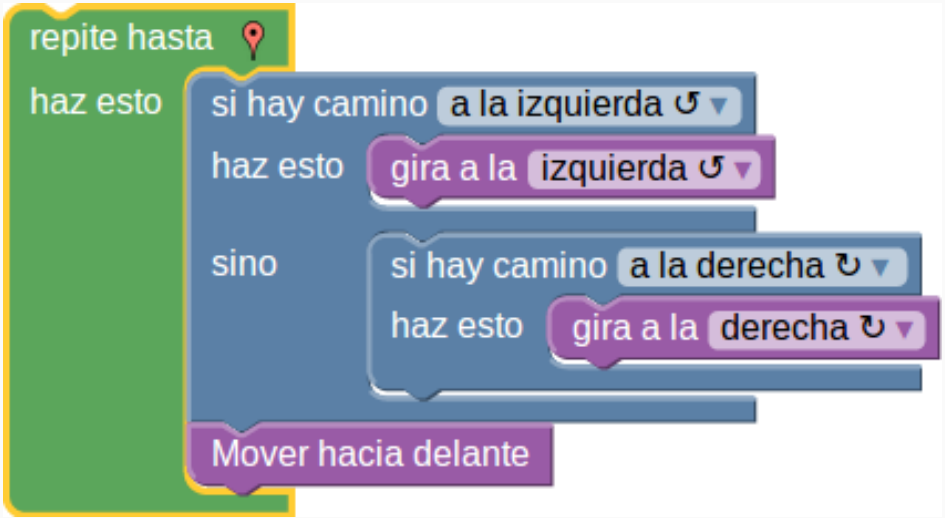
```
1 print("Hello world!")
```

Contenidos

¡Pregunta!

¿Qué hace el siguiente algoritmo?

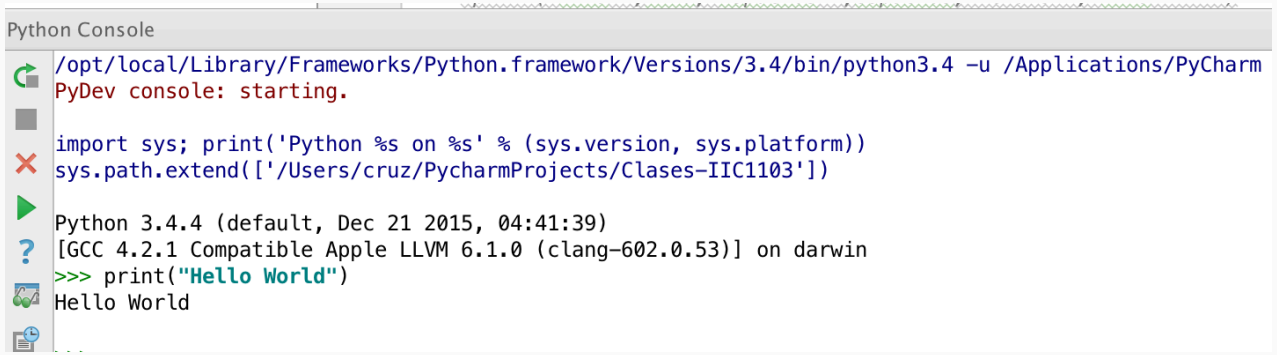




Escribamos un algoritmo

Blockly: https://blockly-demo.appspot.com/static/demos/code/index.html

ó **Python**



Problema 1: Escribir en pantalla

Problema 1: Cree un programa que escriba en pantalla el mensaje "*Alerta de emergencia. Mensaje de prueba sin costo*".

Algoritmo:

1. **Escribir** en pantalla el texto: "Alerta de emergencia. Mensaje de prueba sin costo"

Problema 1: Escribir en pantalla

Problema 1: Cree un programa que escriba en pantalla el mensaje "*Alerta de emergencia. Mensaje de prueba sin costo*".


```
1 print("Alerta de emergencia. Mensaje de prueba sin costo
      ")
```

Hemos hecho un programa que **escribe** una **salida** (*output*)

Problema 2: Preguntar, recordar y escribir

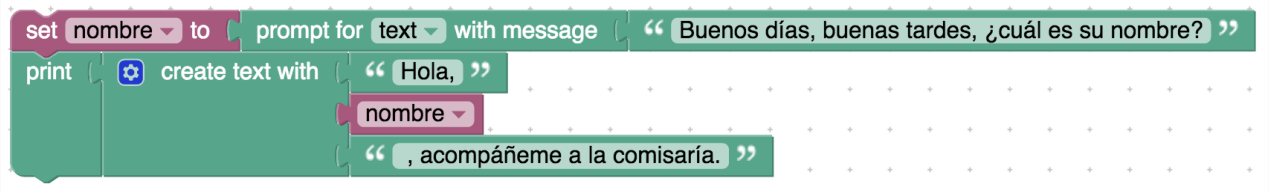
Problema 2: Cree un programa que pregunte al usuario el nombre y lo salude.

Algoritmo:

1. Preguntar el nombre de la persona
2. Recordar el nombre de la persona
3. Escribir el saludo usando el nombre de la persona

Problema 2: Preguntar, recordar y escribir

Problema 2: Cree un programa que pregunte al usuario el nombre y lo salude.




```
1 nombre = input("Buenos días, buenas tardes, ?`cuál es su
      nombre? ")
2 print("Hola," + nombre + ", acompáñeme a la comisaría.")
```

Hemos hecho un programa que **pregunta** un texto de **entrada** (*input*) al usuario, lo **recuerda** (en una *variable*), y luego **escribe** un texto de **salida** (*output*)

Problema 3: Operaciones matemáticas

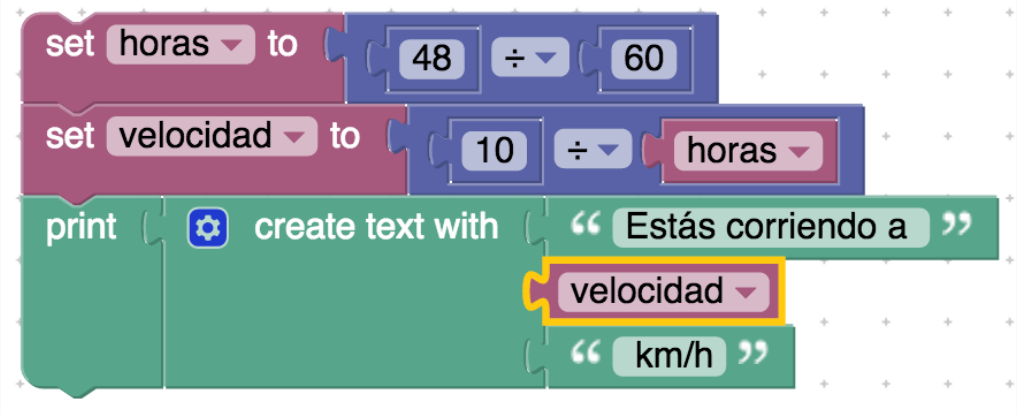
Problema 3: Cree un programa que calcule mi velocidad en km/h, si demoro 48 minutos en correr 10 kilómetros.

Algoritmo:

1. Convertir los minutos a horas.
2. Dividir los kilómetros por las horas.
3. Escribir el resultado de la división.

Problema 3: Operaciones matemáticas

Problema 3: Cree un programa que calcule mi velocidad en *km/h*, si demoro 48 minutos en correr 10 kilómetros.




```
1 horas = 48/60
2 velocidad = 10 / horas
3 print("Estás corriendo a " + str(velocidad) + " km/h")
```

Hemos hecho un programa que **calcula** un valor usando **operaciones matemáticas**, y **escribe** el resultado en un texto de **salida** (*output*)

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a San Joaquín.

Algoritmo:

1. Preguntar el color de la ruta actual
2. Si la ruta actual es roja, entonces debo subirme
3. Si la ruta es verde, entonces no debo subirme
4. Si no hay ningún color, entonces puedo subirme

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a San Joaquín.


```
1 color = input("¿De qué color es la ruta?")
2 if color.lower() == "rojo":
3     print("¡Súbete!, pero deja bajar antes de subir.")
4 elif color.lower() == "verde":
5     print("No te subas. No para en San Joaquín.")
6 else:
7     print("¡Súbete!, pero deja bajar antes de subir.")
```

Problema 4: Condicionales

```
1 color = input("¿De qué color es la ruta?")
2 if color.lower() == "rojo":
3     print("¡Súbete!, pero deja bajar antes de subir.")
4 elif color.lower() == "verde":
5     print("No te subas. No para en San Joaquín.")
6 else:
7     print("¡Súbete!, pero deja bajar antes de subir.")
```

Hemos hecho un programa que ejecuta *una acción u otra* (i.e. toma una **decisión**), dependiendo de una **condición lógica** (*booleana*).

Problema 5: Repeticiones (iteraciones)

Problema 5: Voy en el metro hacia Baquedano. Cree un programa que me avise si debo bajarme del metro.

Algoritmo:

1. Preguntar la estación actual
2. Mientras no sea Baquedano:
    2.1 No debo bajarme
    2.2 Volver a preguntar la estación
3. Si es Baquedano, indicar que debo bajar.

Problema 5: Repeticiones (iteraciones)

Problema 5: Voy en el metro hacia Baquedano. Cree un programa que me avise si debo bajarme del metro.


```
1 estacion = input("¿En qué estación estás?")
2 while estacion.lower() != "baquedano":
3     print("Aún no has llegado. ¡No te bajes!")
4     estacion = input("¿En qué estación estás?")
5 print("Has llegado a Baquedano. ¡Bájate!")
```

Hemos hecho un programa que **repite** una (o más) acción(es), dependiendo de una **condición lógica** (*booleana*).

¡Y eso es!

TODOS los programas que vamos a escribir (y que existen) se pueden construir con:

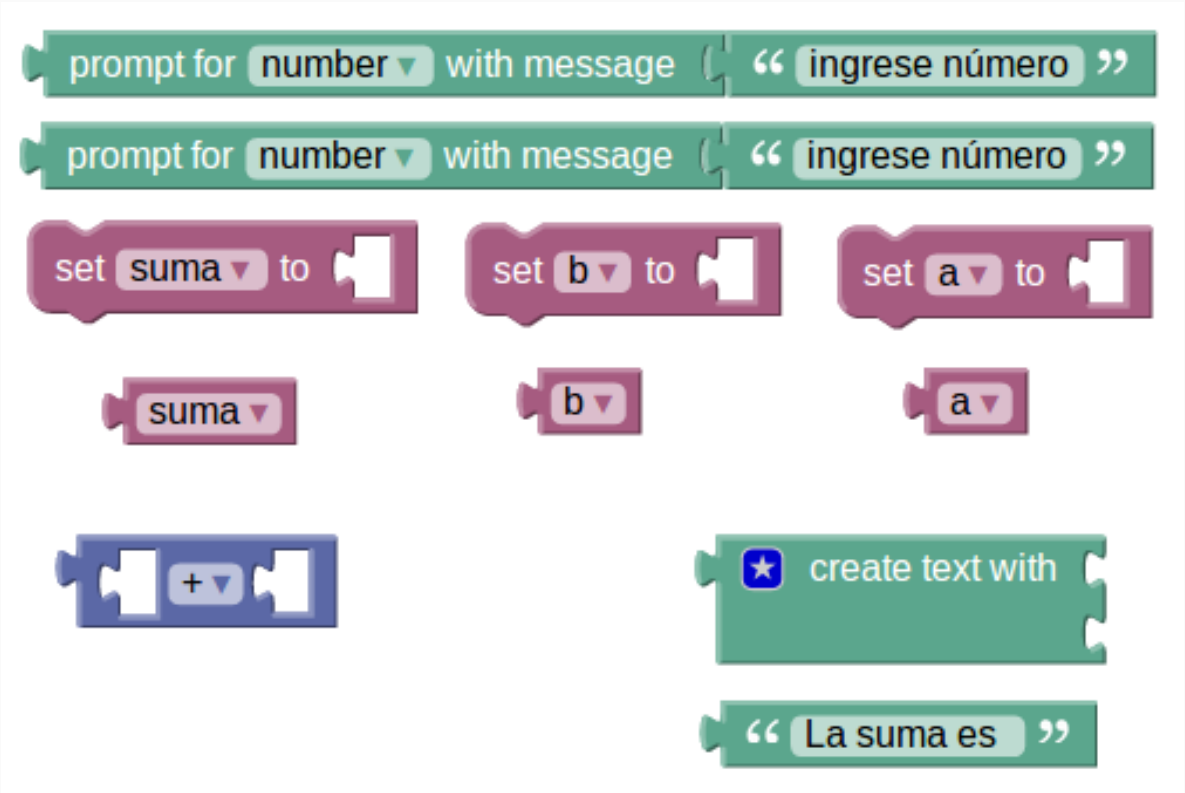
- Datos de *entrada* (*input*) que se *leen*
- Datos de *salida* (*output*) que se *escriben*
- **Variables** que *recuerdan* datos.
- **Operaciones matemáticas** que *calculan* datos.
- Instrucciones **condicionales** que se ejecutan dependiendo de una condición.
- Instrucciones **repetitivas** que se ejecuten múltiples dependiendo de una condición.

Contenidos

Actividad #2A: Sumando dos números

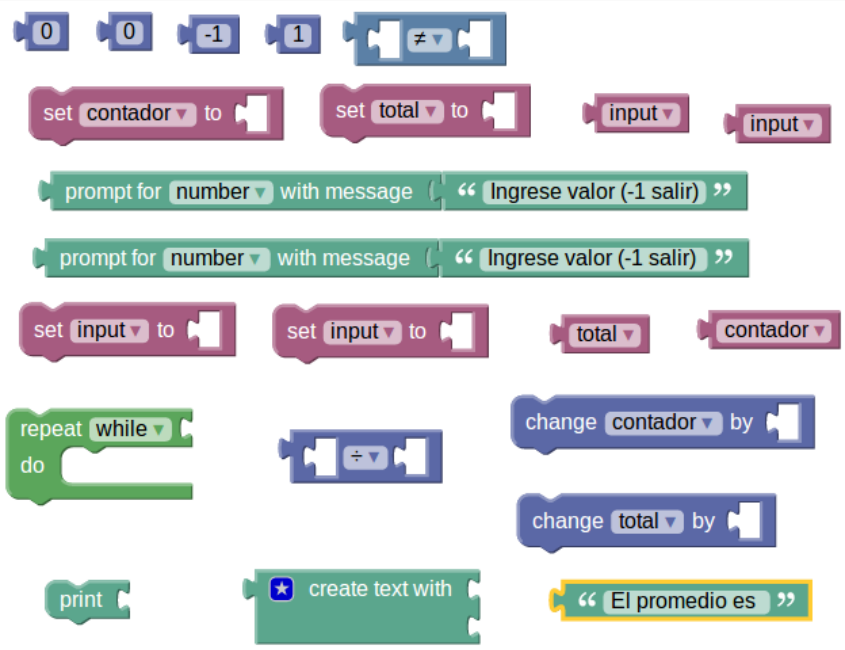
Cree un programa que pida dos números al usuario, los sume y luego los muestre en pantalla. Use los bloques presentes en este **link**.

`https://blockly-demo.appspot.com/static/apps/code/index.html?lang=en#e3coyz`



Actividad #2B: Calculando un promedio

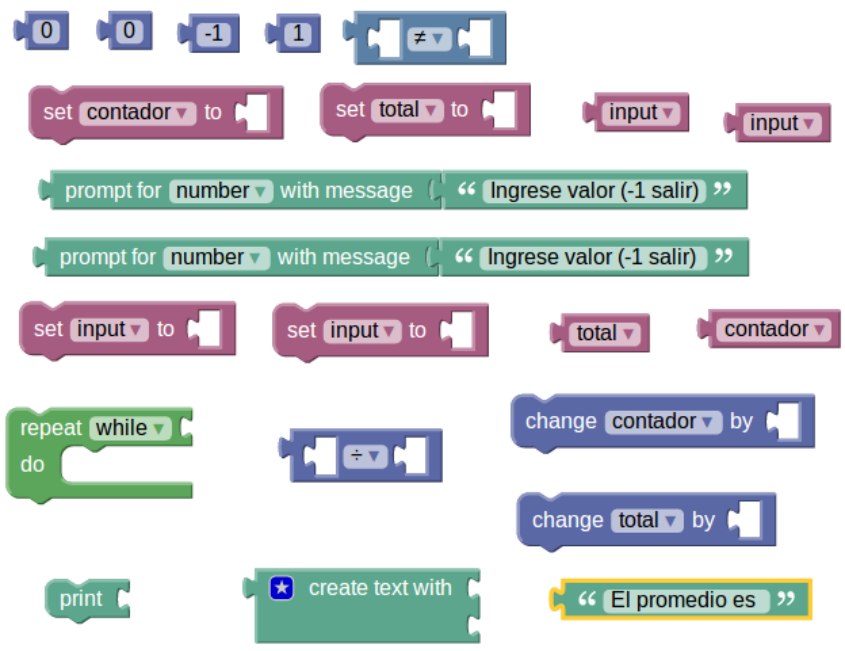
Cree un programa que calcule el promedio de los números ingresados por el usuario. Mientras no se ingrese un -1, el programa debe seguir pidiendo datos. Cuando se ingrese un -1, el programa muestra el promedio y finaliza. Use los bloques presentes en este **link**. `https://blockly-demo.appspot.com/static/apps/code/index.html?lang=en#ofbe7h`



Actividad #2C: Promedio y máximo

Extienda el programa que calcula el promedio para calcular también el número más grande ingresado por el usuario (el máximo).

Para esta parte puede utilizar bloques adicionales a los de la parte anterior.



Puede entregar un archivo Python