IIC1103 - Sección 4

Clase 2: Algoritmos

Prof. Francisca Cattan

Universidad Católica de Chile

8 de marzo, 2023

¿Como empezar?

Objetivo:
Aprender a programar.

¿Como empezar?

Objetivo:
Aprender a programar.

Concepto de algoritmo:

- Algoritmo es una secuencia de pasos.
- Programa está **escrito en un lenguaje**, e **implementa un algoritmo**.

¿Cómo programar?

Usando una herramienta apropiada: un computador.

¿Cómo programar?

Usando una herramienta apropiada: un computador.
Al cual le decimos qué queremos hacer con los datos.

¿Cómo programar?

Usando una herramienta apropiada: un computador.
Al cual le decimos qué queremos hacer con los datos.
Especificándole un algoritmo.

¿Cómo programar?

Usando una herramienta apropiada: un computador.
Al cual le decimos qué queremos hacer con los datos.
Especificándole un algoritmo.
¿Cómo le decimos qué hacer? Hablando su lenguaje: lenguaje binario.

Character	Binary Code	Character	Binary Code	Character	Binary Code
A	01000001	a	01100001	!	00100001
B	01000010	b	01100010	"	00100010
C	01000011	c	01100011	#	00100011
D	01000100	d	01100100	$	00100100
E	01000101	e	01100101	%	00100101
F	01000110	f	01100110	&	00100110
G	01000111	g	01100111	'	00100111
H	01001000	h	01101000	(	00101000
I	01001001	i	01101001	)	00101001
J	01001010	j	01101010	*	00101010
K	01001011	k	01101011	+	00101011
L	01001100	l	01101100	,	00101100
M	01001101	m	01101101	-	00101101
N	01001110	n	01101110	.	00101110
O	01001111	o	01101111	/	00101111
P	01010000	p	01110000	0	00110000
Q	01010001	q	01110001	1	00110001
R	01010010	r	01110010	2	00110010
S	01010011	s	01110011	3	00110011
T	01010100	t	01110100	4	00110100
U	01010101	u	01110101	5	00110101
V	01010110	v	01110110	6	00110110
W	01010111	w	01110111	7	00110111
X	01011000	x	01111000	8	00111000
Y	01011001	y	01111001	9	00111001
Z	01011010	z	01111010	?	00111111
				@	01000000

¿Cómo programar?

En general no escribimos programas de esta manera.

- Código binario es un código de **bajo nivel**.
- Es más fácil para un humano escribir en un lenguaje más cercano al lenguaje natural: **lenguaje de alto nivel**.
- Lamentablemente los computadores aún no son capaces de entender lenguaje natural (aún).

Eso es mentira, yo se que se puede programar con una IA



write a code in Python 3 that asks for 2 random numbers, prints the higest and then the sum of both.

Sure! Here's a code snippet that does what you're looking for:

python Copy code

```python
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))

if num1 > num2:
    print("Highest number is:", num1)
else:
    print("Highest number is:", num2)

print("The sum is:", num1 + num2)
```

In this code, we first prompt the user to enter two random numbers using the `input()` function. Since we want to perform arithmetic operations on these numbers, we convert them from strings to integers using the `int()` function.

We then use an `if` statement to compare the two numbers and determine which is the highest. If `num1` is greater than `num2`, we print `num1` as the highest number, otherwise we print `num2`.

Finally, we print the sum of both numbers by adding them together with the `+` operator.

¿Es correcto usar sólo IA para programar?



Is it possible to code in python only using chat gpt and knowing nothing of programming languages?

While ChatGPT can provide you with helpful information and code snippets, it's generally not recommended to try to learn programming solely through conversation with a language model.

Programming requires a foundational understanding of concepts such as variables, control structures, functions, and data types. While these concepts can be explained through text-based instructions and examples, programming is also about developing a way of thinking and problem-solving that can take time and practice to develop.

That being said, if you're interested in learning to code, there are many resources available online to help you get started. There are free interactive tutorials, online courses, and even YouTube videos that can walk you through the basics of programming in Python or any other language you're interested in.

¿Es correcto usar sólo IA para programar?



I'm a python programming teacher. What can I say to my students who want to do all the assignments using only chatgpt.

¿Es correcto usar sólo IA para programar?

If your students want to do all their programming assignments using only ChatGPT, it's important to emphasize that using a language model as the sole means of completing assignments is not a practical or effective approach to learning programming. While ChatGPT can provide guidance and support, it's not a substitute for hands-on practice and the development of programming skills.

You can explain to your students that programming is a practical skill that requires practice, experimentation, and problem-solving. By relying solely on ChatGPT, they miss out on the opportunity to learn through trial and error, and to develop their own unique problem-solving approaches.

Additionally, using ChatGPT to complete assignments may be considered a form of academic dishonesty, as it involves submitting work that was not created solely by the student. This could lead to disciplinary action and negative consequences for their academic career.

Encourage your students to use ChatGPT as a supplement to their learning, but emphasize the importance of hands-on practice and critical thinking to develop their programming skills. Encourage them to work on their own, seek help from peers, and ask questions when they're stuck. These are all important steps towards becoming proficient programmers.

Regenerate response

¿Cómo programar?

¿Qué es un lenguaje de programación?

- Comprensible por el humano (programador) y el computador.

¿Cómo programar?

¿Qué es un lenguaje de programación?

- Comprensible por el humano (programador) y el computador.

Los lenguajes de programación definen y compilan un conjunto de instrucciones para la CPU (Unidad central de procesamiento) para realizar cualquier tarea específica. Cada lenguaje de programación tiene un conjunto de palabras clave junto con la sintaxis que utiliza para crear instrucciones.

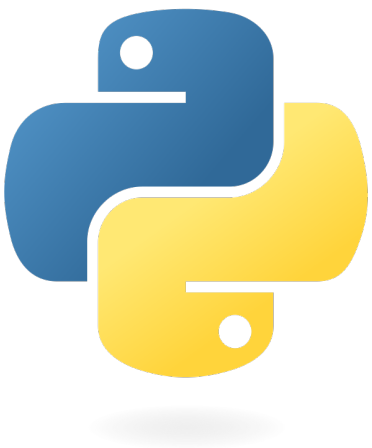
¿Cómo programar?

¿En qué lenguaje?



¿Cómo programar?

¿En qué lenguaje?



python™

¿Cómo programar?

¿Por qué Python?

C++:

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

¿Cómo programar?

¿Por qué Python?

Java:

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

¿Cómo programar?

¿Por qué Python?

Python:

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

La materia de hoy: Algoritmos

Algorithm *(noun.)*

Word used by programmers when...
they do not want to explain what they did.

Algoritmos

RAE:

Conjunto ordenado y finito de operaciones que permite hallar la solución de un problema.

Wikipedia:

Un algoritmo es cualquier cosa que funcione paso a paso, donde cada paso se puede describir sin ambigüedad.

Algoritmos

- Tipo de pensamiento crítico (analítico).

Algoritmos

- Tipo de pensamiento crítico (analítico).
- Modelar (imaginar) y descomponer un problema.
  - Descomposición y Abstracción.

Algoritmos

- Tipo de pensamiento crítico (analítico).
- Modelar (imaginar) y descomponer un problema.
  - Descomposición y Abstracción.
- Preveer una secuencia de pasos para resolver el problema
  - Crear algoritmos

Algoritmos

- Tipo de pensamiento crítico (analítico).
- Modelar (imaginar) y descomponer un problema.
  - Descomposición y Abstracción.
- Preveer una secuencia de pasos para resolver el problema
  - Crear algoritmos

- Un buen entendimiento o conceptualización de un problema ayuda a resolverlo mejor.
- Un buen algoritmo se programa o implementa mejor.

Algoritmos

Pensamiento Computacional:

- Descomponer: dividir el problema en partes más pequeñas y manejables.[1]

[1] divide y vencerás

Algoritmos

Pensamiento Computacional:

- Descomponer: dividir el problema en partes más pequeñas y manejables.[1]
- Reconocer Patrones: buscar similitudes entre las distintas partes y problemas que ya han solucionado.

[1] divide y vencerás

Algoritmos

Pensamiento Computacional:

- Descomponer: dividir el problema en partes más pequeñas y manejables.[1]
- Reconocer Patrones: buscar similitudes entre las distintas partes y problemas que ya han solucionado.
- Abstraerse: Conectarse en lo importante e ignorar lo irrelevante.

[1] divide y vencerás

Algoritmos

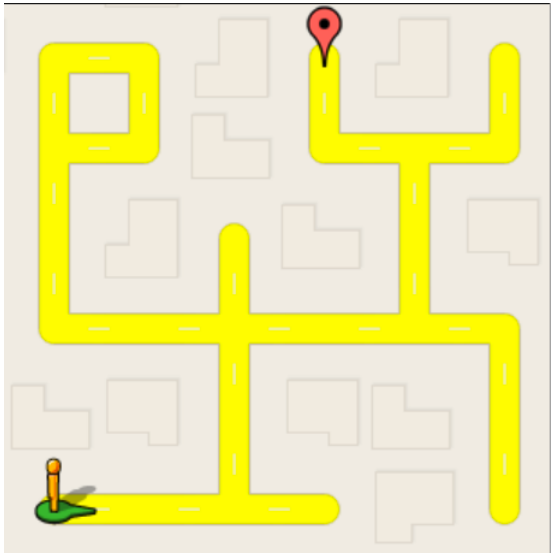
Pensamiento Computacional:

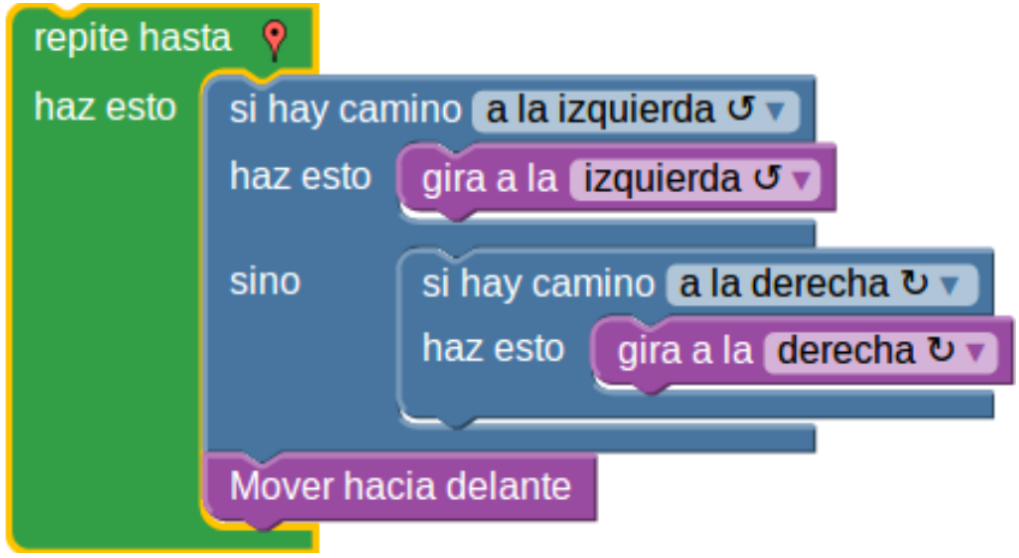
- Descomponer: dividir el problema en partes más pequeñas y manejables.[1]
- Reconocer Patrones: buscar similitudes entre las distintas partes y problemas que ya han solucionado.
- Abstraerse: Conectarse en lo importante e ignorar lo irrelevante.
- Algoritmo: Diseñar pasos o reglas que solucionen cada parte del problema.

[1] divide y vencerás

Algoritmos

¿Qué hace el siguiente algoritmo?





Problema 1: Escribir en pantalla

Problema 1: Cree un programa que escriba en pantalla el mensaje *"Sistema de inscripción de cursos UC. Ingrese su número de alumno."*

Problema 1: Escribir en pantalla

Problema 1: Cree un programa que escriba en pantalla el mensaje *"Sistema de inscripción de cursos UC. Ingrese su número de alumno."*

Algoritmo:

1. **Escribir** en pantalla el texto: "Sistema de inscripción de cursos UC. Ingrese su número de alumno."

Problema 1: Escribir en pantalla

Problema 1: Cree un programa que escriba en pantalla el mensaje *"Sistema de inscripción de cursos UC. Ingrese su número de alumno."*

```
1 print("Sistema de inscripcion de cursos UC. Ingrese su
  numero de alumno.")
```

Problema 1: Escribir en pantalla

Problema 1: Cree un programa que escriba en pantalla el mensaje *"Sistema de inscripción de cursos UC. Ingrese su número de alumno."*

```
1 print("Sistema de inscripcion de cursos UC. Ingrese su
  numero de alumno.")
```

Hemos hecho un programa que **escribe** una **salida**.

Problema 2: Preguntar, recordar y escribir

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

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.

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.

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 dias, ¿cual es su nombre? ")
2 print("Hola, " + nombre + "un gusto conocerle.")
```

Problema 2: Preguntar, recordar y escribir

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

```
1 nombre = input("Buenos dias, ¿cual es su nombre? ")
2 print("Hola, " + nombre + "un gusto conocerle.")
```

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 recorrer 10 kilómetros. Luego, muestra el resultado.

Problema 3: Operaciones matemáticas

Problema 3: Cree un programa que calcule mi velocidad en km/h, si demoro 48 minutos en recorrer 10 kilómetros. Luego, muestra el resultado.

Algoritmo:

1. **Convertir** los minutos a hora.

Problema 3: Operaciones matemáticas

Problema 3: Cree un programa que calcule mi velocidad en km/h, si demoro 48 minutos en recorrer 10 kilómetros. Luego, muestra el resultado.

Algoritmo:

1. **Convertir** los minutos a hora.
2. **Dividir** los kilómetros por hora.

Problema 3: Operaciones matemáticas

Problema 3: Cree un programa que calcule mi velocidad en km/h, si demoro 48 minutos en recorrer 10 kilómetros. Luego, muestra el resultado.

Algoritmo:

1. **Convertir** los minutos a hora.
2. **Dividir** los kilómetros por hora.
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 recorrer 10 kilómetros. Luego, muestra el resultado.

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

Problema 3: Operaciones matemáticas

Problema 3: Cree un programa que calcule mi velocidad en km/h, si demoro 48 minutos en recorrer 10 kilómetros. Luego, muestra el resultado.

```
1 horas = 48/60
2 velocidad = 10/horas
3 print("Estas 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 Parque Bustamante, siendo las 8:20AM en día de semana.

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a Parque Bustamante, siendo las 8:20AM en día de semana.

Algoritmo:

1. **Preguntar** el color de la ruta actual.

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a Parque Bustamante, siendo las 8:20AM en día de semana.

Algoritmo:

1. **Preguntar** el color de la ruta actual.
2. **Si** la ruta actual es roja, entonces debo subirme.

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a Parque Bustamante, siendo las 8:20AM en día de semana.

Algoritmo:

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

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a Parque Bustamante, siendo las 8:20AM en día de semana.

Algoritmo:

1. **Preguntar** el color de la ruta actual.
2. **Si** la ruta actual es roja, entonces debo subirme.
3. **Si** la ruta actual 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 Parque Bustamante, siendo las 8:20AM en día de semana.

```
color = input("De que color es la ruta?")
if color.lower() == "rojo":
    print("Subete!, pero deja bajar antes de subir.")
elif color.lower() == "verde":
    print("No te subas. No para en Parque Bustamante.")
else:
    print("Subete!, pero deja bajar antes de subir.")
```

Problema 4: Condicionales

Problema 4: Cree un programa que me ayude a saber si debo subirme al metro o no para llegar a Parque Bustamante, siendo las 8:20AM en día de semana.

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

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

Problema 5: Repeticiones o iteraciones

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

Problema 5: Repeticiones o iteraciones

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

Algoritmo:

1. **Preguntar** la estación actual.

Problema 5: Repeticiones o iteraciones

Problema 5: Voy en metro hacia estación 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.

Problema 5: Repeticiones o iteraciones

Problema 5: Voy en metro hacia estación 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 o iteraciones

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

```
1 estacion = input("En que estacion estas?")
2 while estacion.lower() != "baquedano":
3     print("Aun no has llegado. No te bajes.")
4     estacion = input("En que estacion estas?")
5 print("Has llegado a estacion Baquedano. Bajate.")
```

Problema 5: Repeticiones o iteraciones

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

```
1 estacion = input("En que estacion estas?")
2 while estacion.lower() != "baquedano":
3     print("Aun no has llegado. No te bajes.")
4     estacion = input("En que estacion estas?")
5 print("Has llegado a estacion Baquedano. Bajate.")
```

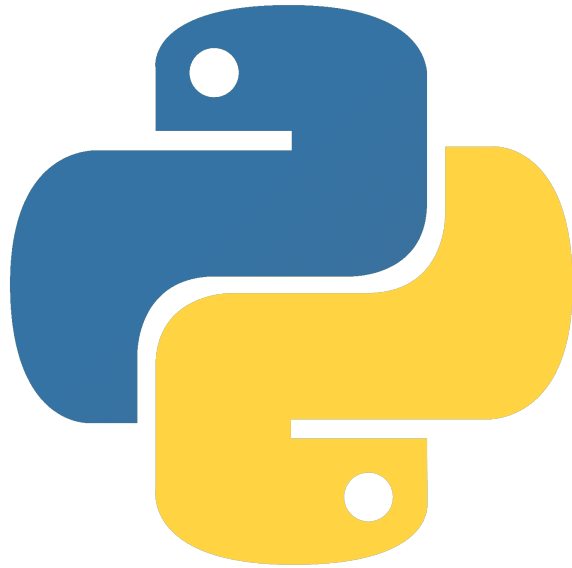
Hemos hecho un programa que **repite** una (o mas) accion(es) dependiendo de una **condición lógica** (booleana).

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 ejecutan múltiples veces dependiendo de una condición.

¿Programemos?



python™