

Todas y Todos podemos programar

Academias de invierno UOH

Contenidos

Segunda clase programación

¿Qué frases entienden los computadores?

- a. Sumar, restar, multiplicar y dividir con Python.
- b. ¿Cómo representa la información mi computador?
- c. Explicación del concepto variable.

Mi computador es el mejor para el juego de la verdad.

- d. Comparaciones numéricas (mayor, menor, igual) en Python.
- e. Comparaciones y manejo de palabras en Python.

¿Qué frases entienden los computadores?

Operaciones matemáticas

- En matemáticas, un **operador** es un símbolo que se usa cuando resolvemos un problema matemático.

- + (sumar)
- - (restar)
- x (multiplicar)
- ÷ (division)
- // (division entera)
- % (modulo)
- Exp (exponencial)

```
[ 3 ] 1 # suma 13  
      2 6 + 7 + 4 + 5
```

22

```
[ ] 1 # resta  
    2 10 - 5
```

5

```
[ ] 1 # multiplicacion 18  
    2 3 * 6
```

18

```
▶ 1 # division  
  2 8 / 3
```

▶ 2.6666666666666665

```
[ ] 1 # division entera  
    2 12//3
```

4

```
▶ 1 # modulo  
  2 10 % 5
```

▶ 0

```
[ 4 ] 1 # exponencial 2 * 2 * 2  
      2 5 ** 3
```

125

¿Qué frases entienden los computadores?

¿Cómo representa la información mi computador?

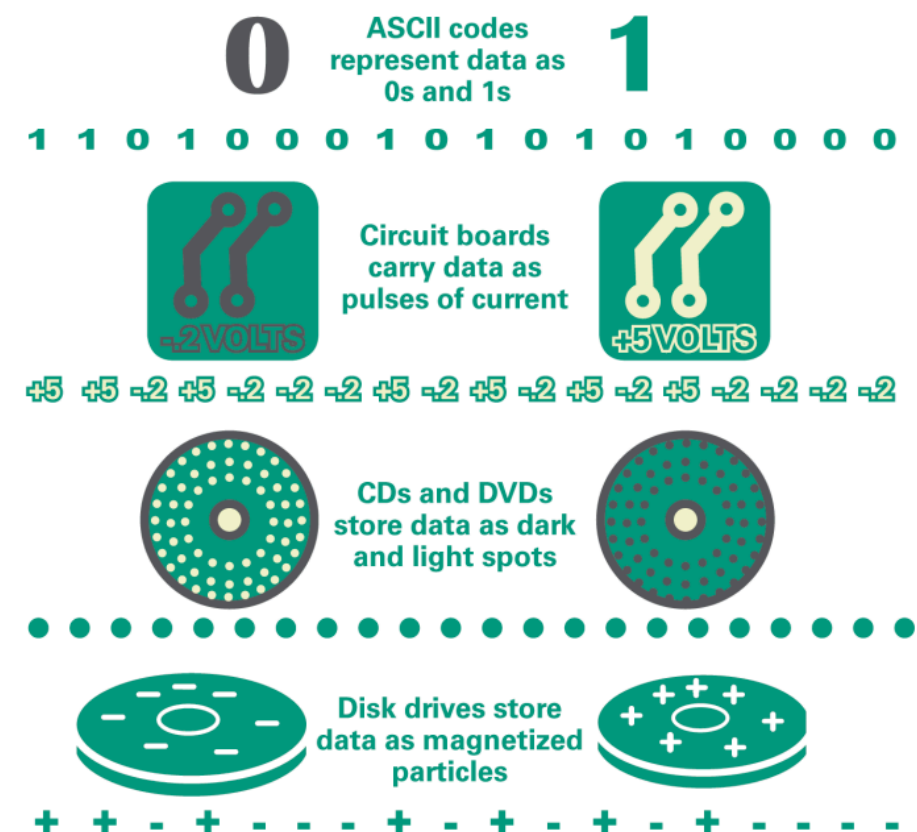
- **Los datos** se refieren a los símbolos que representan a las personas, hechos, cosas e ideas. Los datos pueden ser un nombre, un número, los colores en una fotografía, o las notas en una composición musical.
- La representación de datos se refiere a la forma en que los datos se **almacenan, procesan y transmiten**.
- Celulares y las computadoras **almacenan datos en formatos digitales** que pueden ser manejados por **circuitos electrónicos**.
- La **digitalización** es el proceso de convertir información, como texto, números, fotos o música, en datos digitales que pueden ser manipulados por dispositivos electrónicos.



¿Qué frases entienden los computadores?

¿Cómo representa la información mi computador?

- Los 0 y 1 utilizados para representar datos digitales son denominados dígitos binarios — de este término obtenemos la palabra **bit** que significa **dígito binario**.
- Un **bit** es un 0 o un 1 usado en la representación digital de datos.
- Un archivo digital (archivo), es una colección de datos con nombre que existe en un medio de almacenamiento, como un **disco duro, CD, DVD o unidad flash**.



¿Qué frases entienden los computadores?

Representando números

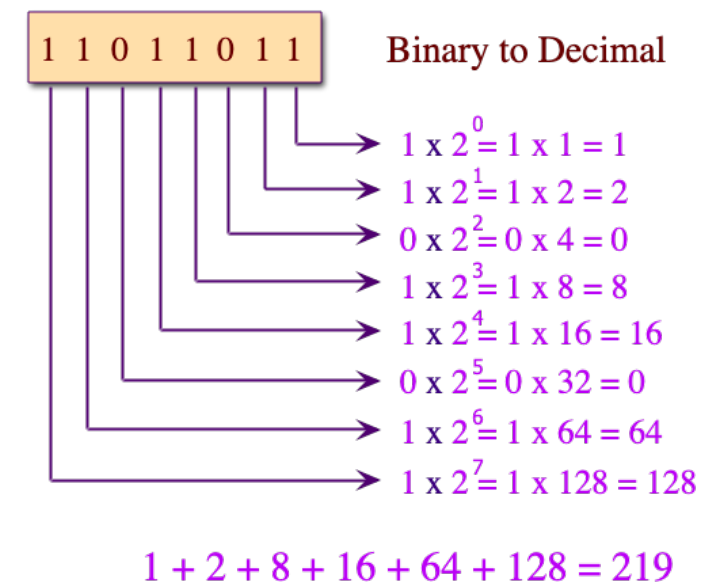
- Los **computadores** representan datos numéricos usando el sistema numérico binario, también llamado **base 2**
- El sistema numérico binario solo tiene dos dígitos: **0 y 1**.
- En python: usamos las funciones **bin** y **int** para convertir números a binario y de binarios a números.

```
0 -> 0b0 -> 0
1 -> 0b1 -> 1
2 -> 0b10 -> 2
3 -> 0b11 -> 3
4 -> 0b100 -> 4
5 -> 0b101 -> 5
6 -> 0b110 -> 6
7 -> 0b111 -> 7
8 -> 0b1000 -> 8
9 -> 0b1001 -> 9
10 -> 0b1010 -> 10
11 -> 0b1011 -> 11
12 -> 0b1100 -> 12
13 -> 0b1101 -> 13
14 -> 0b1110 -> 14
15 -> 0b1111 -> 15
16 -> 0b10000 -> 16
17 -> 0b10001 -> 17
18 -> 0b10010 -> 18
19 -> 0b10011 -> 19
219 -> 0b11011011 -> 219
```

```
a = bin(5)
print(a)
print(int(a))
```

ASCII extendido utiliza ocho bits para cada carácter (2^8 caracteres).

A es 01000001



¿Qué frases entienden los computadores?

Representando texto

ASCII extendido utiliza ocho bits para cada carácter (2^8 caracteres).

A es 01000001

Tabla ASCII

00100000	Space	00110011	3	01000110	F	01011001	Y	01101100	l
00100001	!	00110100	4	01000111	G	01011010	Z	01101101	m
00100010	"	00110101	5	01001000	H	01011011	[01101110	n
00100011	#	00110110	6	01001001	I	01011100	\	01101111	o
00100100	\$	00110111	7	01001010	J	01011101]	01110000	p
00100101	%	00111000	8	01001011	K	01011110	^	01110001	q
00100110	&	00111001	9	01001100	L	01011111	_	01110010	r
00100111	'	00111010	:	01001101	M	01100000	`	01110011	s
00101000	(00111011	;	01001110	N	01100001	a	01110100	t
00101001)	00111100	<	01001111	O	01100010	b	01110101	u
00101010	*	00111101	=	01010000	P	01100011	c	01110110	v
00101011	+	00111110	>	01010001	Q	01100100	d	01110111	w
00101100	,	00111111	?	01010010	R	01100101	e	01111000	x
00101101	-	01000000	@	01010011	S	01100110	f	01111001	y
00101110	.	01000001	A	01010100	T	01100111	g	01111010	z
00101111	/	01000010	B	01010101	U	01101000	h	01111011	{
00110000	0	01000011	C	01010110	V	01101001	i	01111100	
00110001	1	01000100	D	01010111	W	01101010	j	01111101	}
00110010	2	01000101	E	01011000	X	01101011	k	01111110	~

Python:

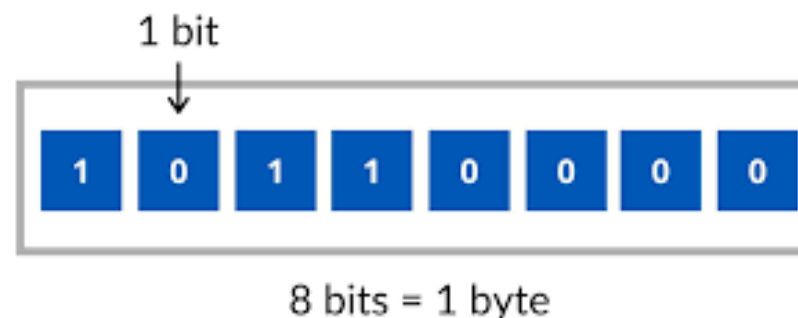
```
#texto a binario
print("A")
print(ord("A"))
print(bin(ord("A")))
print(chr(ord("A")))
```

A
65
0b1000001
A

¿Qué frases entienden los computadores?

Bits y bites

Todos los datos almacenados y transmitidos por computadores se codifican como **bits**.



UNIDAD	SÍMBOLO	EQUIVALENCIA		
Kilobyte	kB	2^{10} bytes	1024 bytes	
Megabyte	MB	2^{10} kilobytes	1024 kilobytes	
Gigabyte	GB	2^{10} megabytes	1024 megabytes	1000.000.000 bytes
Terabyte	TB	2^{10} gigabytes	1024 gigabytes	1 billón de bytes
<u>Petabyte</u>	PB	2^{10} terabytes	1024 terabytes	1000 billones de bytes
Exabyte	EB	2^{10} <u>petabytes</u>	1024 <u>petabytes</u>	1 trillón de bytes
<u>Zettabyte</u>	ZB	2^{10} exabytes	1024 exabytes	1000 trillones de bytes
<u>Yottabyte</u>	YB	2^{10} <u>zettabytes</u>	1024 <u>zettabyte</u>	1 cuatrillón de bytes

¿Qué frases entienden los computadores?

Explicación del concepto variable.

Las variables tiene dos partes:

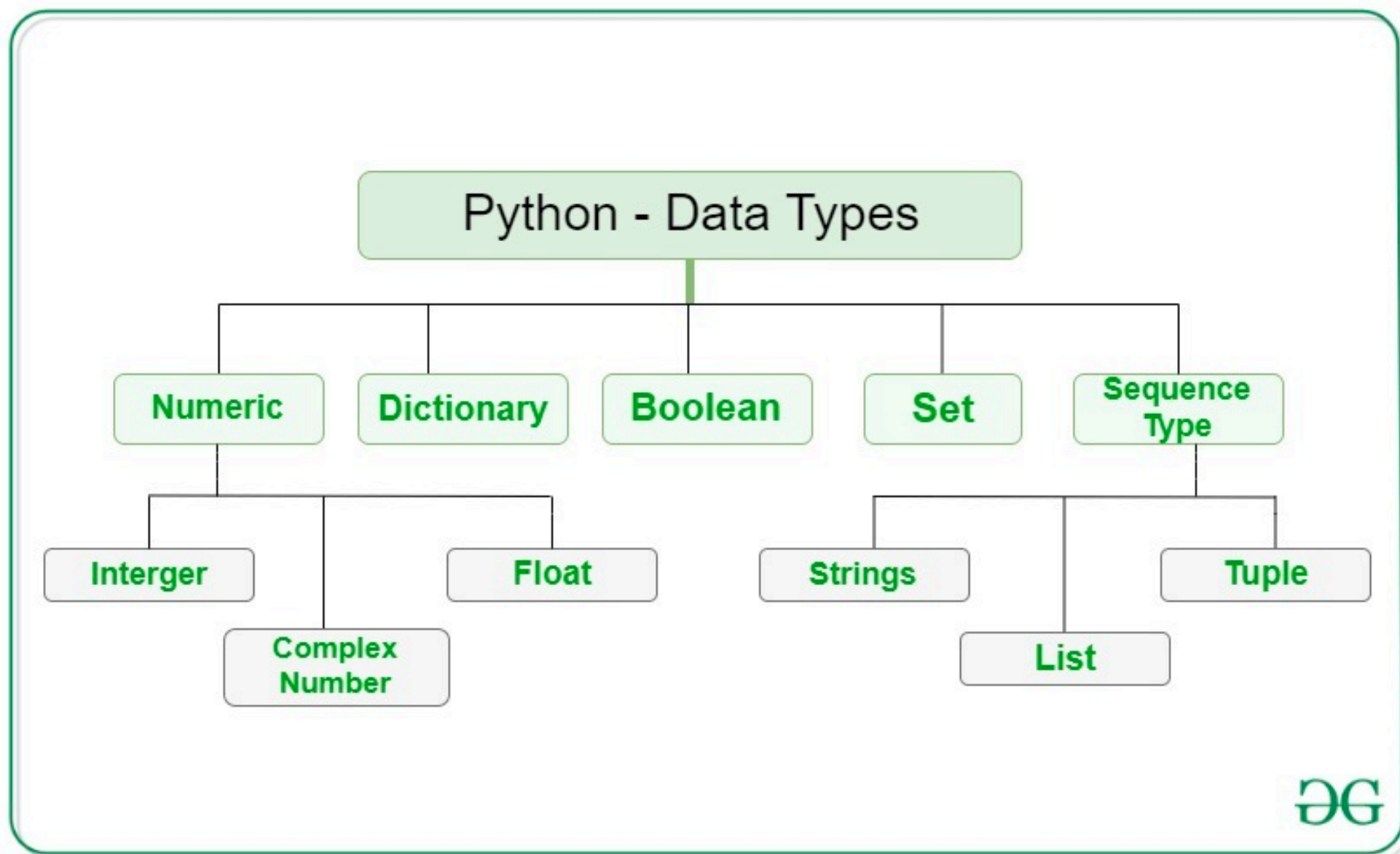
1. **El nombre.** También llamado identificador: algo que se usa para identificar (decir qué es algo). El nombre de la variable se utiliza para identificar un dato exacto.
 2. **El valor.** Valor significa cantidad o tipo. La parte de valor de una variable muestra los datos que necesitamos para realizar un seguimiento.
- Usamos un signo igual (=) para asignar (dar) un **valor** al **nombre** de una **variable**.

```
# variables
a = 5
print(a)
a = 7
print(a)
b = a
print(b)
c = "texto"
b = c
print(b)
```

¿Qué frases entienden los computadores?

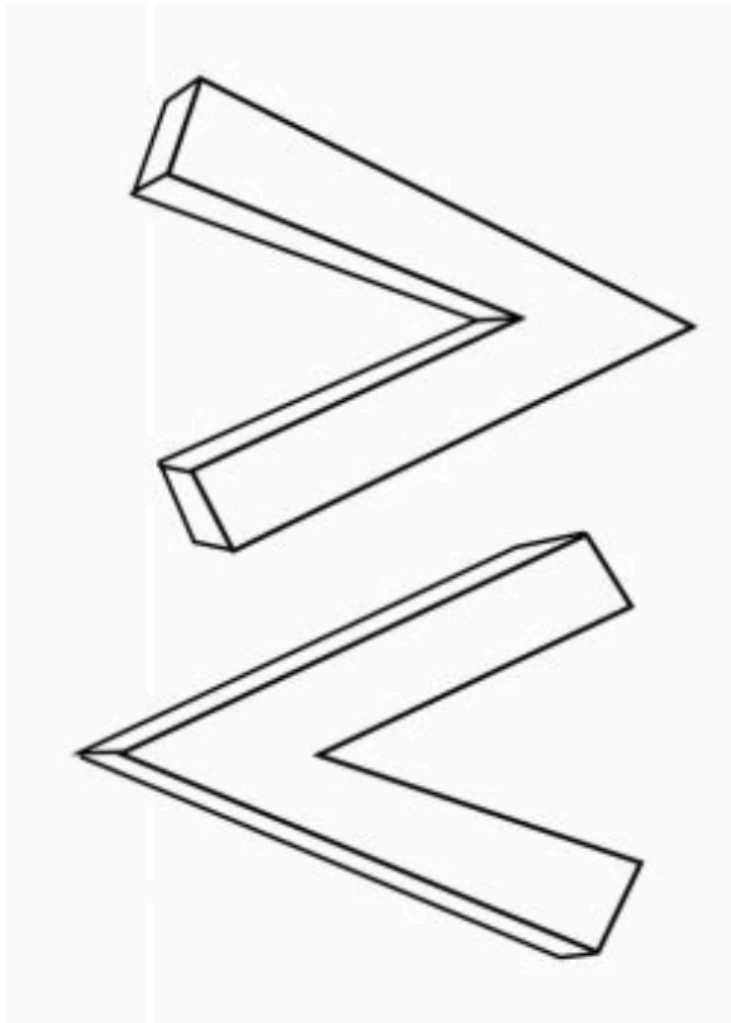
Explicación del concepto variable.

Tipos de variables en python:



Mi computador es el mejor para el juego de la verdad

Comparaciones numéricas (mayor, menor, igual) en Python.



```
1 5 > 7
```

False

```
1 5 < 7
```

True

```
1 a = 100
2 b = 50
3 a < b
```

False

```
1 5 == 5
```

True

```
[66] 1 5 == 6
```

False

```
[67] 1 5 is 5
```

True

```
1 5 is 6
```

False

```
1 5 > 3 and 10 < 20
```

True

```
1 a = 5
2 a is 5 or a == 4
```

True

```
1 a >= 5
```

True

```
1 a <= 10
```

True

```
1 a = "Alex"
2 b = "Carol"
3 a != b
```

True

```
1 a is not b
```

True

Mi computador es el mejor para el juego de la verdad

Manejo y comparaciones de palabras en Python.

- Las palabras (strings) se utilizan para almacenar texto.
- En python se implementan como secuencias. Una secuencia es una colección/arreglo ordenado por posición de objetos idénticos (orden de izquierda-derecha).
- Strings son secuencias de un carácter (listas, tuplas).

Strings contienen índices por posición.

```
[1] 1 S = 'palabra'
     2 len(S)
```

7

```
[7] 1 print(S[0]) #primer caracter de la secuencia
     2 print(S[3]) # cuarto caracter
```

p
a

```
▶ 1 # indices de secuencias comienzan en 0 .. n-1
   2 print(S[-1]) # ultimo caracter
   3 print(S[-2]) # ante-penultimo caracter
```

a
r

Mi computador es el mejor para el juego de la verdad

Manejo y comparaciones de palabras en Python.

- Podemos extraer porciones de un string [pos1:pos2).

```
print(S[2:5])    lab
print(S[1:])     alabra
print(S[:4])     pala
print(S[:-1])    palabr
print(S[:])      palabra
```

- Concatenar strings

```
S + ' dos '
```

- Repetir strings

```
S * 4
```

- Los string son inmutables.

```
S[0] = "P"      S = 'P' + S[1:]
print(S)        Palabra
```


Mi computador es el mejor para el juego de la verdad

Manejo y comparaciones de palabras en Python.

- Los string poseen funciones específicas.

capitalize casefold center count encode **endswith** expandtabs **find**
format format_map index isalnum isalpha **isascii** **isdecimal**
isdigit
isidentifier islower isnumeric isprintable isspace istitle
isupper **join**
ljust lower lstrip maketrans partition replace rfind rindex
rjust rpartition rsplit rstrip **split** splitlines **startswith** strip
swapcase title translate **upper** zfill

```
print(S.capitalize())  
print(S.upper())  
print(S.find("bra"))  
print(S.replace("bra", "BRA"))  
l="aaa,bbb,ccc,ddd\n"  
print(l.split(", "))  
print(l.rstrip())  
print(l.rstrip().split(", "))
```

```
Palabra  
PALABRA  
4  
PalaBRA  
['aaa', 'bbb', 'ccc', 'ddd\n']  
aaa,bbb,ccc,ddd  
['aaa', 'bbb', 'ccc', 'ddd']
```

Una serpiente que programa?

Google Colab

The screenshot shows a Google Colab notebook interface. The browser's address bar displays the URL: `colab.research.google.com/drive/1ZhflFGWy-EI2V-fG1ip1wwBFuHjx7RTM#scrollTo=4cyzIFHpRbnL`. The notebook title is `TPP_C01.ipynb`. The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with options like Comment, Share, and a settings icon. A RAM and Disk usage indicator shows 0% usage. The notebook content is organized into sections with expandable/collapsible icons:

- Todas y todos podemos programar**
Este taller busca entregar competencias basicas de programación a estudiantes de tercero y Cuarto Medio.
- Hello world en python**
Code cell [2]: `1 print("hello world")`
Output: `hello world`
- Obteniendo información**
Code cell [3]: `1 !python --version`
Output: `Python 3.7.13`
- Creando mi primera variable**
Code cell [4]: `1 mensaje = "hello world"`

The status bar at the bottom indicates the notebook is completed at 3:57 AM. The file manager at the bottom shows several files: `DB-W1-C01-Intr....pdf`, `clases (1).zip`, `MainText_revis....docx`, and `Tarea2.ipynb`.

Una serpiente que programa?

GitHub

The screenshot displays the GitHub interface for the repository `adigenova/tpp`. The repository is public and has 1 branch and 0 tags. The commit history shows a recent commit by `adigenova` titled "adding class 1" with 6 commits. The repository structure includes folders `clases`, `code`, and `programa`, and a `README.md` file. The README content is as follows:

Todas y Todos Podemos Programar (TTP)

Curso Introductorio de Python para alumnos de Tercero y cuarto Medio.

Programa del curso

Programa que detalla el contenido de cinco clases teoricas y cinco talleres prácticos.

Bitacora de clases

Archivo guía con el contenido y código publicados en el repositorio github del taller.

The right sidebar shows the repository's statistics: 0 stars, 1 watching, and 0 forks. The 'About' section describes the repository as a "Curso Introductorio de Python para alumnos de Tercero y cuarto Medio". The 'Releases' and 'Packages' sections indicate no releases or packages have been published. The 'Languages' section shows that the repository is 100.0% Jupyter Notebook.

<https://github.com/adigenova/tpp>

Información

- Laboratorio Práctico
 - 11:30 a 13:00
 - Sala A 505
 - 2 ayudantes para el taller.
- Google Colab y mi primer programa en python

Preguntas?
Muchas gracias.