

Manual Usuario

PRACTICA 1

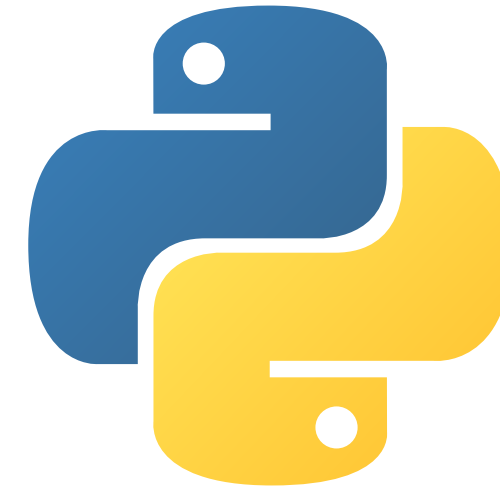
202001954

ESTUARDO SEBASTIÁN VALLE BANCES

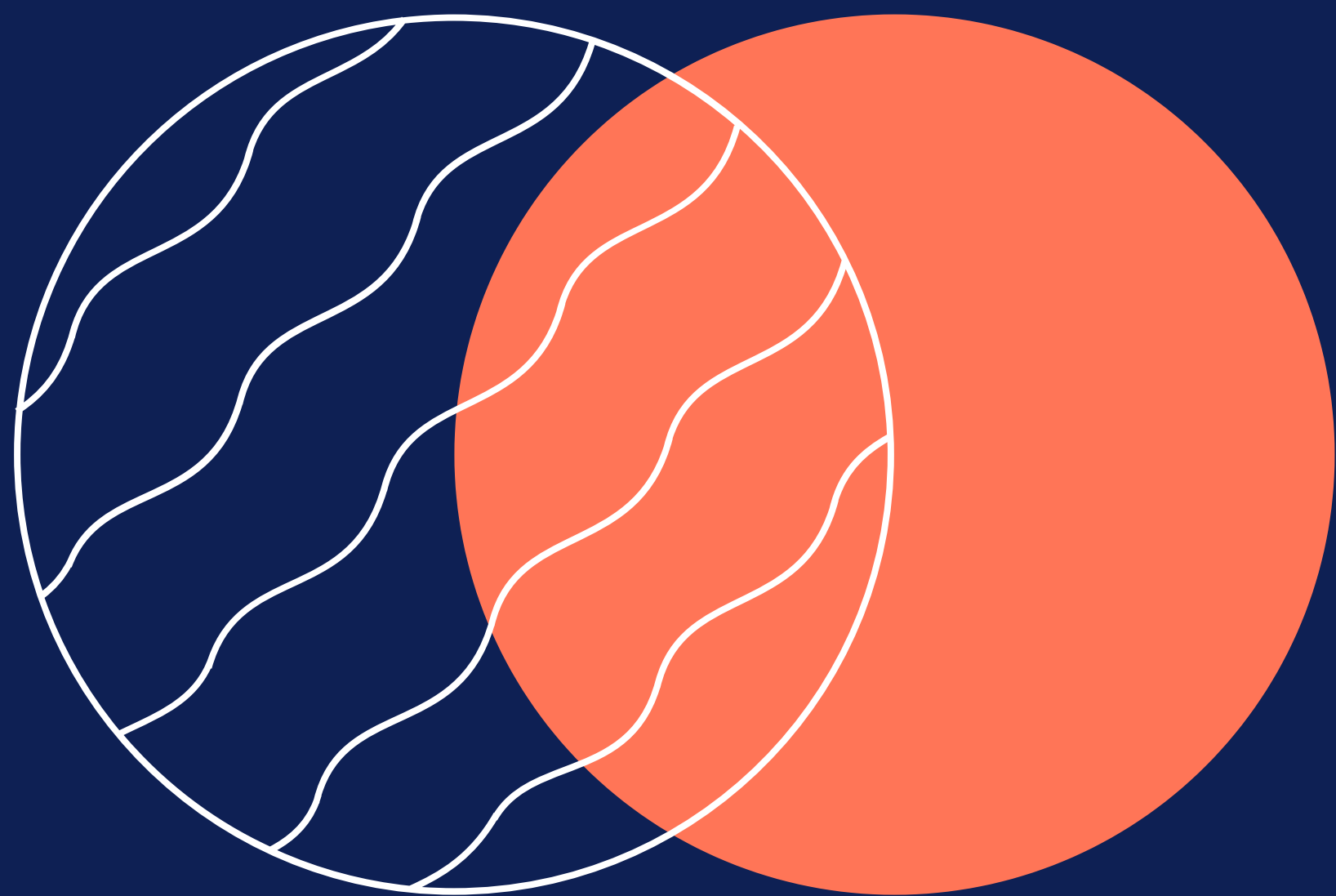


LENGUAJES FORMALES
Y DE PROGRAMACION

Introducción



La presente práctica busco el entendimiento del Lenguaje de Programación "Python" e introdujo conceptos iniciales sobre teoría de Lenguajes y Autómatas. El programa posee una serie de pasos en los cuales se analiza un texto y se crean salidas y reportes con CSS y HTML.



Lenguajes de Entrada

Productos

En el lenguaje de entrada:

NOMBREMES deberá ser una cadena

AÑO deberá ser un entero

Los productos serán declarados cómo:

["Nombre del Prod", precio, unidad]

```
NOMBRE_MES : AÑO = (  
    ["producto 1", 25, 33];  
    ["producto 2", 35.75, 10];  
    ["producto 3", 15, 170];  
    ["producto 4", 50, ];  
)
```

Instrucciones

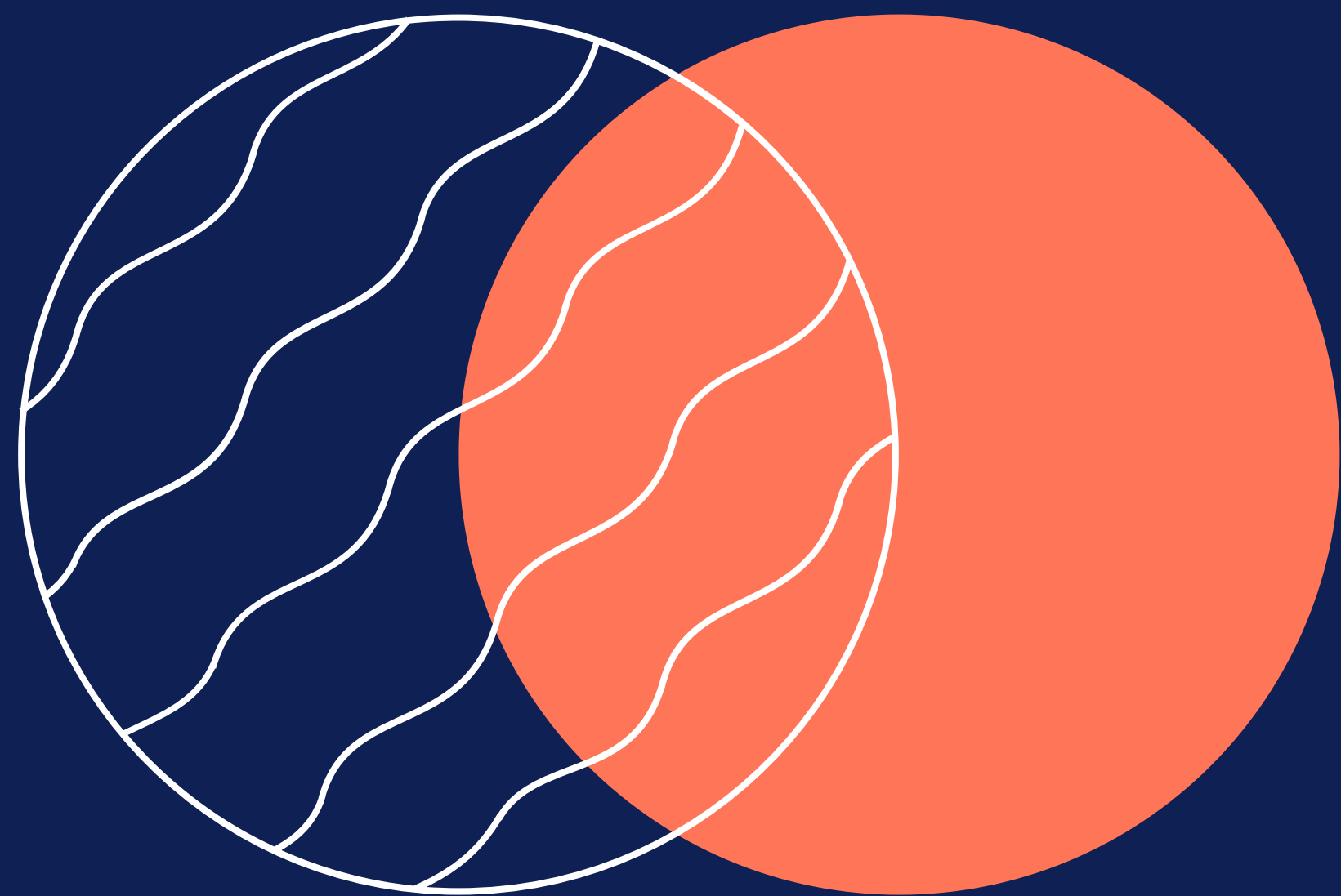
```
<?
  Nombre: "reporte01",
  Grafica: "Barras",
  Titulo: "Reporte de Ventas Agosto",
  TituloX: "Producto",
  TituloY: "Total"
?>
```

En el lenguaje de entrada posee características similares a las de un archivo JSON, sin embargo, este no necesitará de un orden pre-establecido.

NOMBRE indicará el nombre del archivo png

GRAFICA indicará el tipo de grafica

TITULO, TITULOX y TITULOY los titulos de los respectivos ejes



Diseño

Identificación de Caracteres

El programa basa su interpretación en el concepto de "Lookahead". Es decir, existen ciertos caracteres que determinan el guardado de cadenas acumuladas.

```
NOMBRE_MES : AÑO = (  
  [\"producto 1\" 25 33] ;  
  [\"producto 2\" 35.75 10] ;  
  [\"producto 3\" 15 170] ;  
  [\"producto 4\" 50, ] ;  
  .  
)
```


Inicialización del Programa

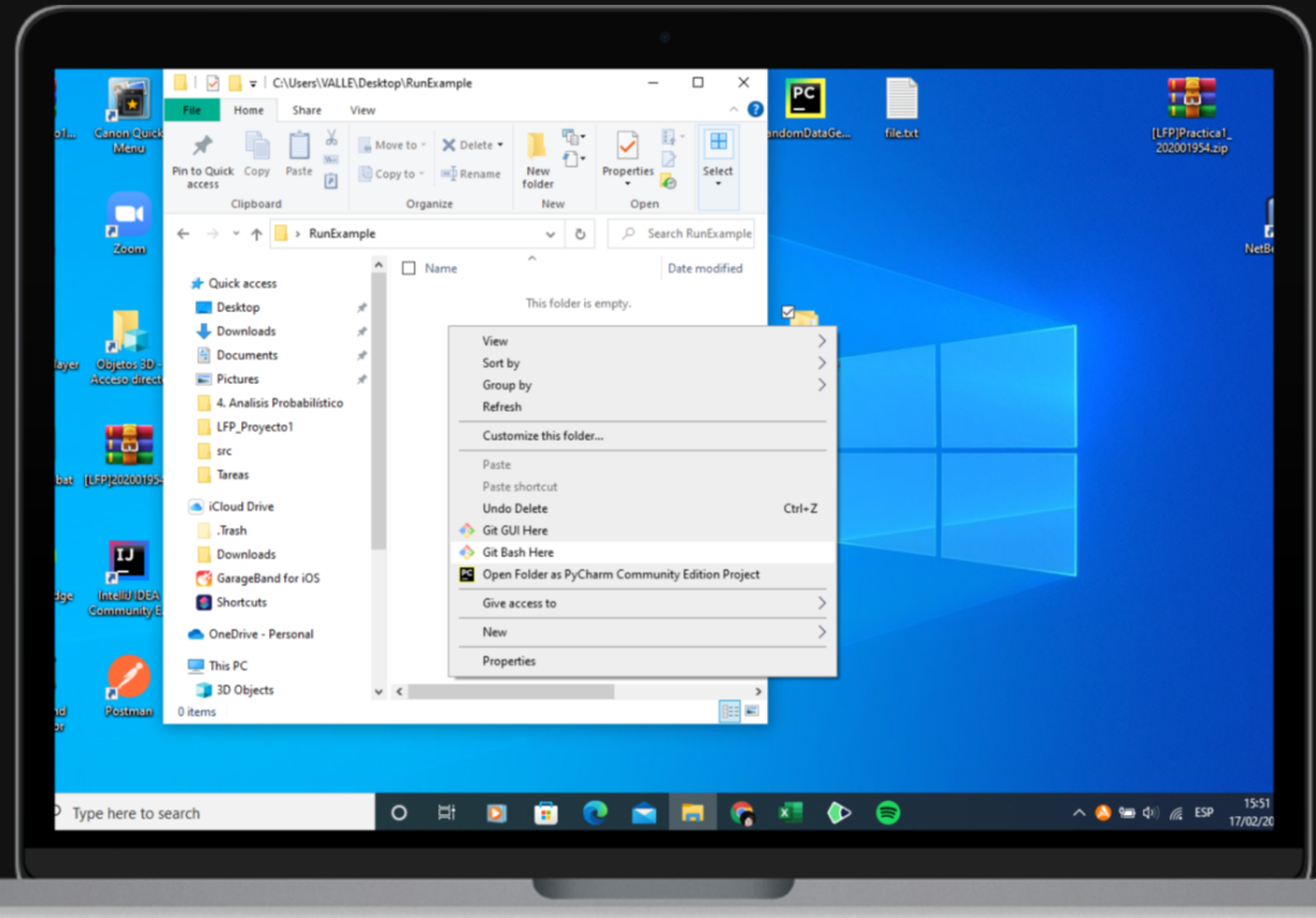


Clonar Repositorio

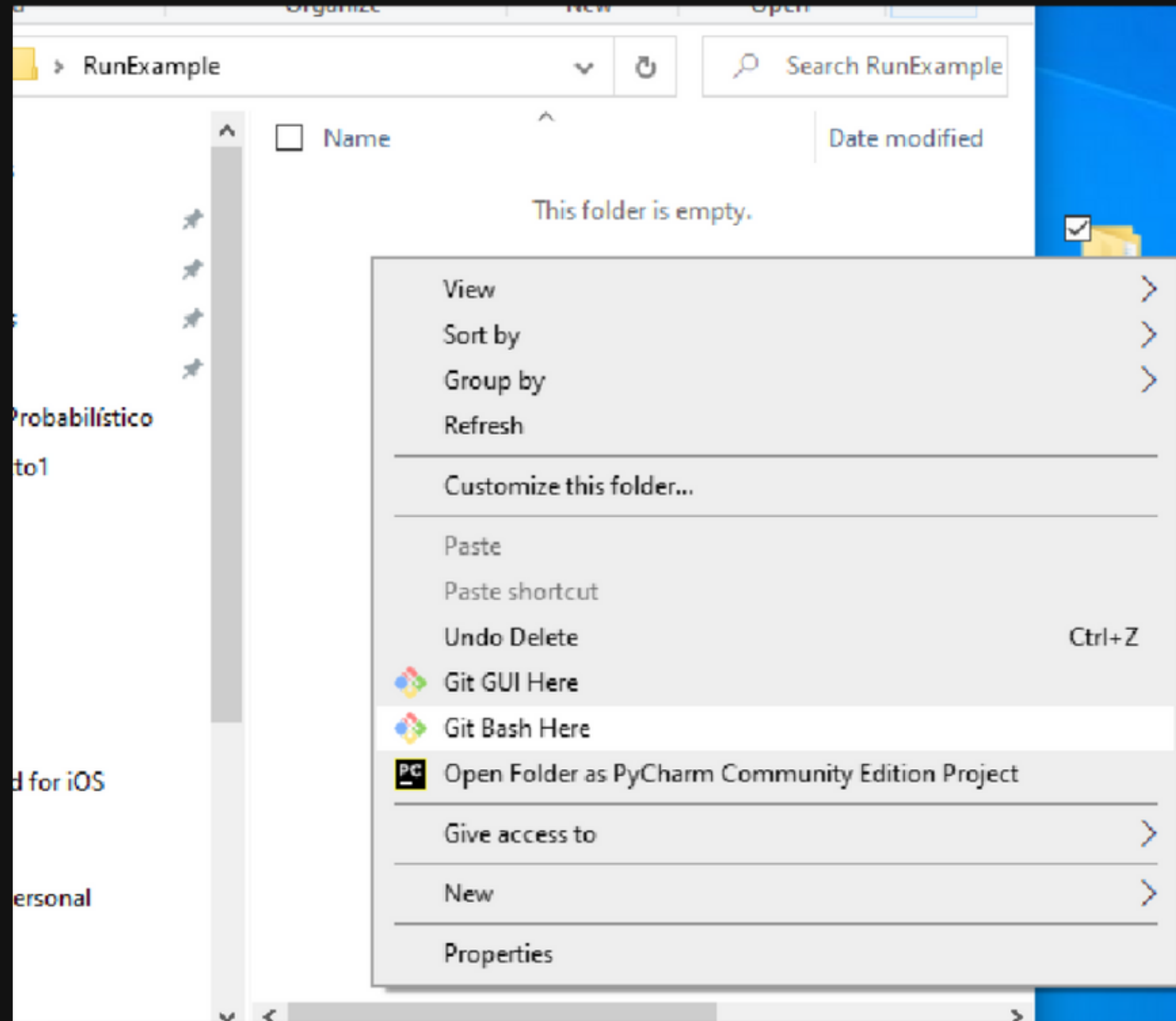
En primera instancia es necesario adquirir el código a partir de GITHUB

(Si usted no posee la herramienta GIT, puede obtenerla en el siguiente link

<https://git-scm.com/downloads>



Clonar Repositorio

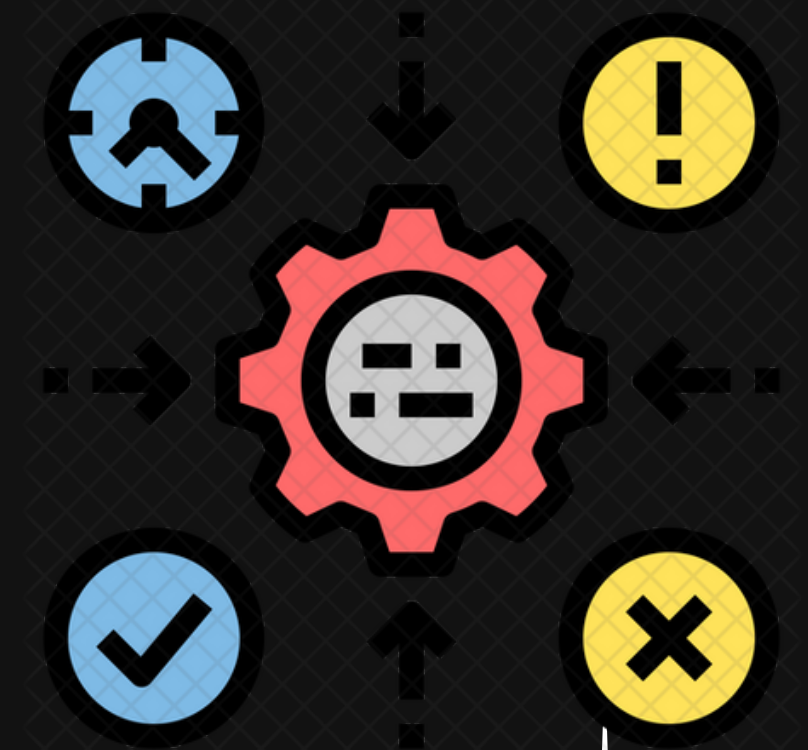


```
MINGW64:/c/Users/VALLE/Desktop/RunExample
VALLE@DESKTOP-S1KEU8S MINGW64 ~/Desktop/RunExample
$ git clone https://github.com/Vallit0/LFP_PR_202001954/tree/main/src
```

Es necesario utilizar el comando GIT BASH para abrir la consola de GIT

Posteriormente, solicitar el clonado del repositorio con

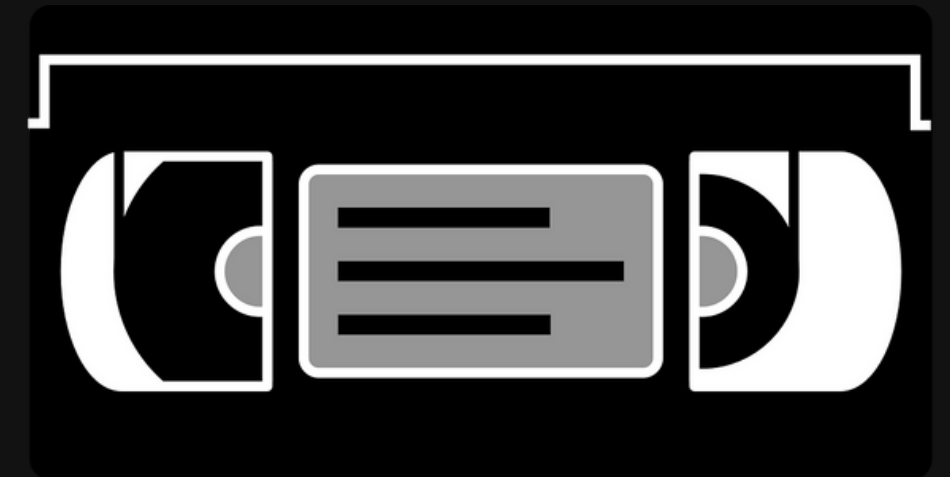
GIT CLONE link proporcionado



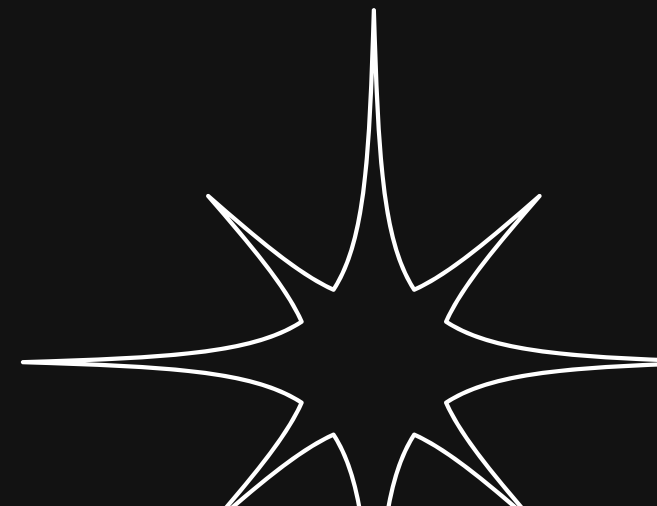
Clonar Repositorio

```
MINGW64:/c/Users/VALLE/Desktop/RunExample
VALLE@DESKTOP-S1KEU8S MINGW64 ~/Desktop/RunExample
$ git clone https://github.com/Vallit0/LFP_PR_202001954
Cloning into 'LFP_PR_202001954'...
remote: Enumerating objects: 67, done.
remote: Counting objects: 100% (67/67), done.
remote: Compressing objects: 100% (52/52), done.
Remote: Total 67 (delta 25), reused 46 (delta 11), pack-reused 0
Receiving objects: 100% (67/67), 169.68 KiB | 597.00 KiB/s, done.
Resolving deltas: 100% (25/25), done.

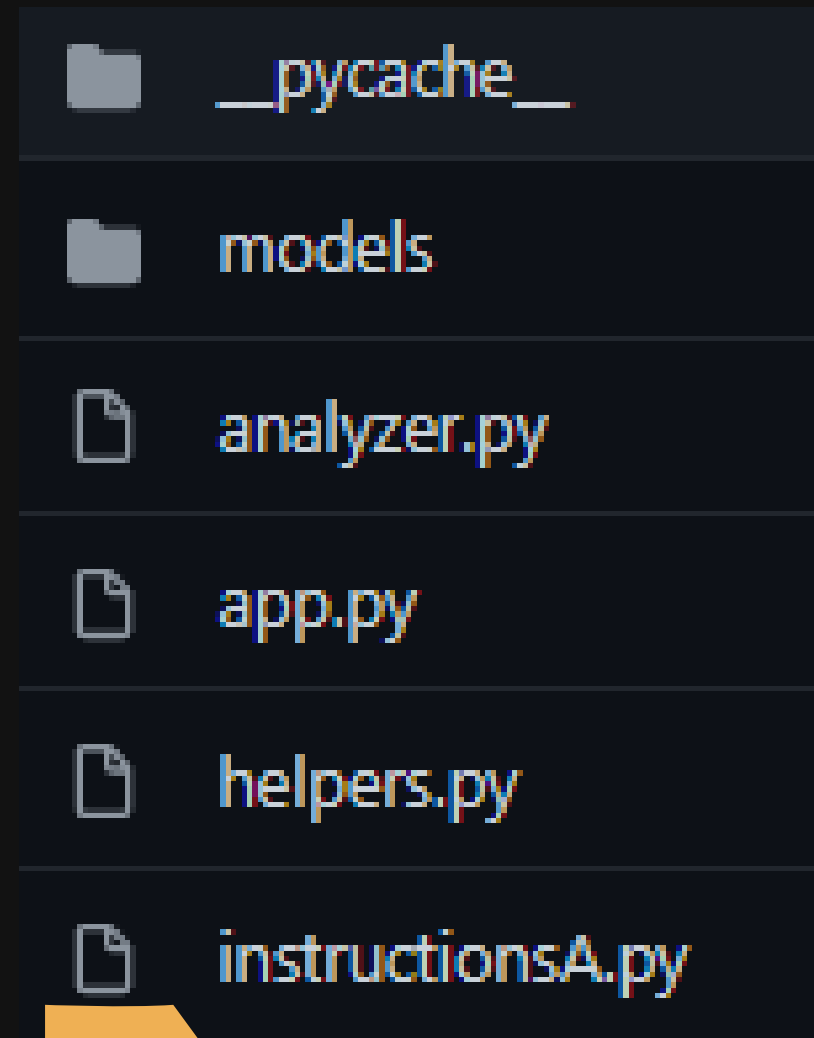
VALLE@DESKTOP-S1KEU8S MINGW64 ~/Desktop/RunExample
$ |
```



Media vez haya sido clonado el repositorio, el siguiente mensaje será mostrado en consola



Source



A screenshot of a Windows File Explorer window showing the directory structure of a project. The path is `RunExample > LFP_PR_202001954 > src`. The table below represents the content of the window.

<input type="checkbox"/>	Name	Date modified	Type
	__pycache__	17/02/2022 15:55	File folder
	models	17/02/2022 15:55	File folder
	analyzer.py	17/02/2022 15:55	JetBrains PyChar...
	app.py	17/02/2022 15:55	JetBrains PyChar...
	helpers.py	17/02/2022 15:55	JetBrains PyChar...
	instructionsA.py	17/02/2022 15:55	JetBrains PyChar...

Es así cómo el código, los archivos .py y demás deberán aparecer en la carpeta elegida para el repositorio



Source

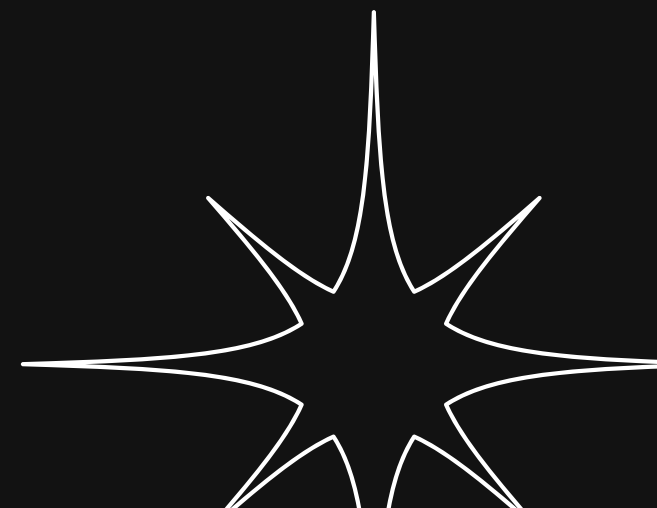
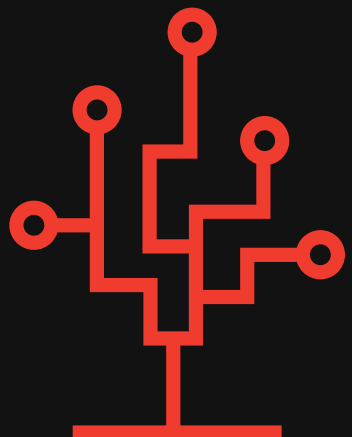
 ➤ RunExample ➤ LFP_PR_202001954 ➤ src

```
C:\Windows\System32\cmd.exe  
Microsoft Windows [Version 10.0.19042.746]  
(c) 2020 Microsoft Corporation. All rights reserved.  
C:\Users\VALLE\Desktop\RunExample\LFP_PR_202001954\src>
```

Esta consola deberá ser abierta desde la carpeta, esto puede realizarse escribiendo cmd desde la pestaña



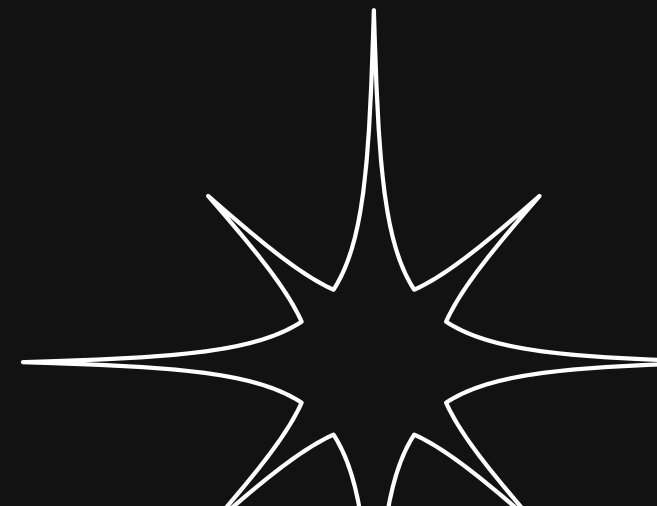
Para correr el programa será necesario ingresar el comando "Python app.py" en consola.

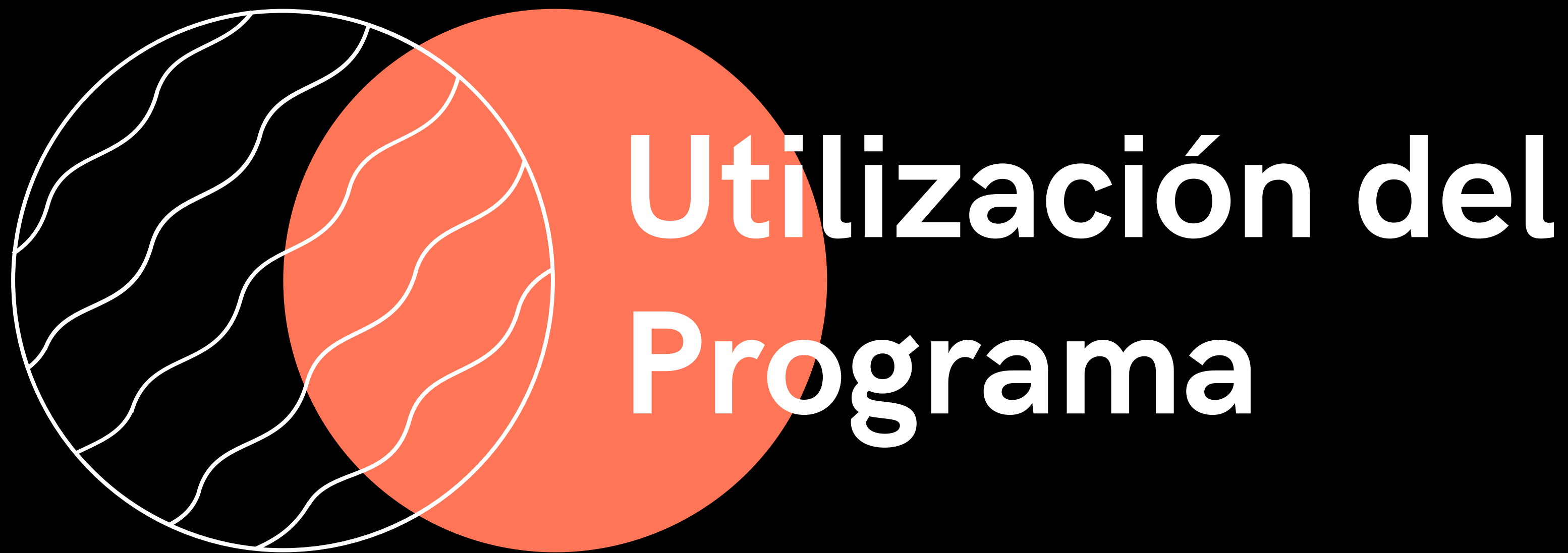


Source

```
C:\Users\VALLE\Desktop\RunExample\LFP_PR_202001954\src>python app.py
=====Menu =====
| 1. Cargar Data |
| 2. Cargar Instrucciones |
| 3. Analizar |
| 4. Reporte |
| 5. Salir |
=====
Ingresa una opción:
```

Es así cómo se mostrará el menu principal

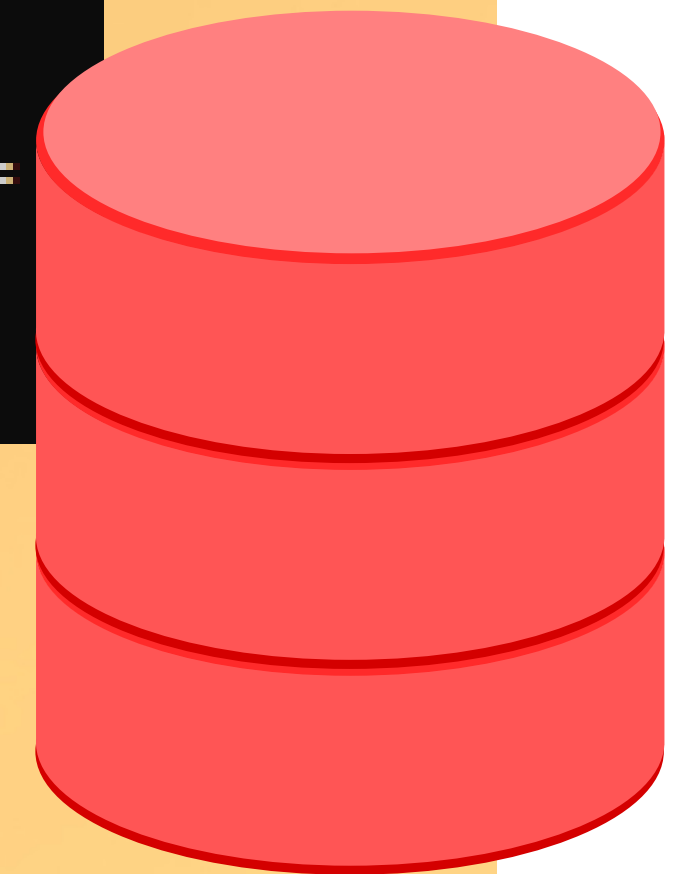




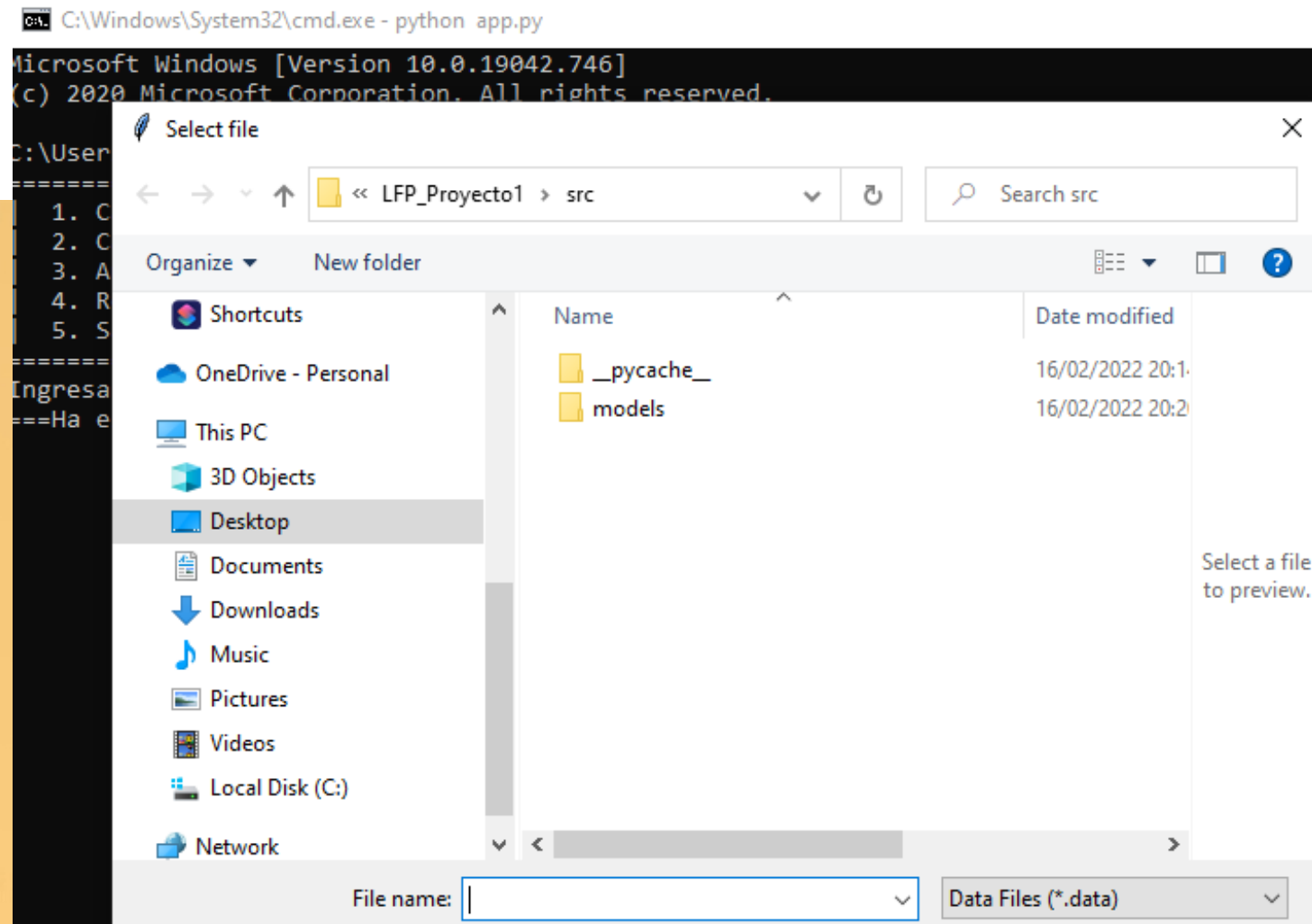
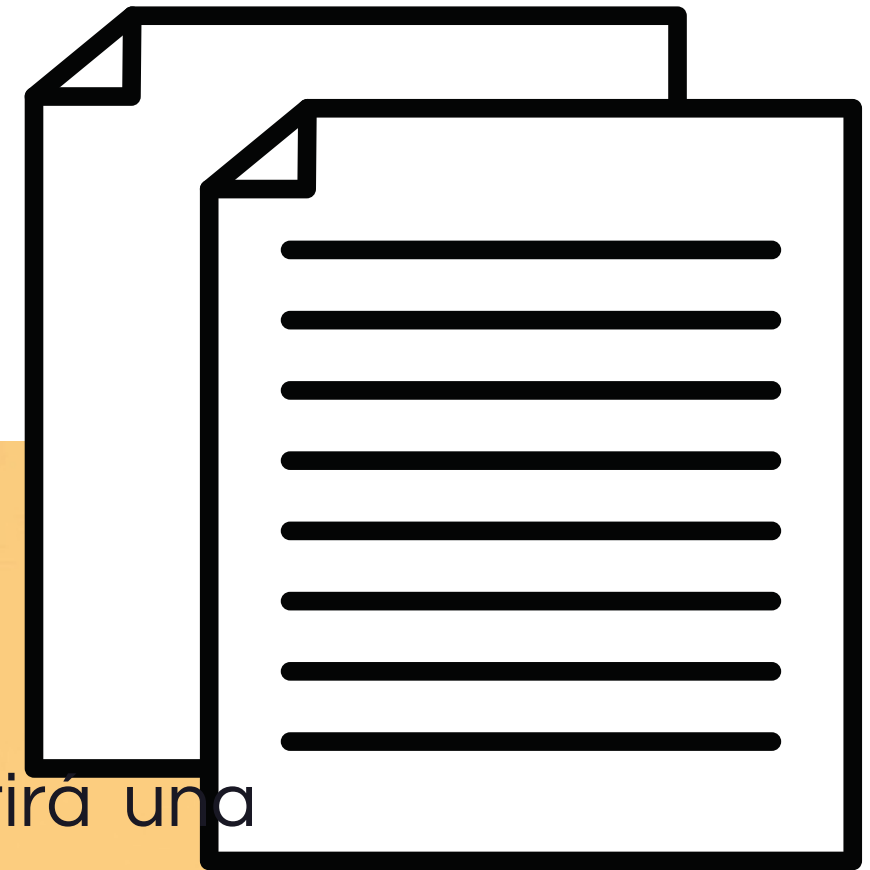
Menu

```
=====Menu=====
| 1. Cargar Data |
| 2. Cargar Instrucciones |
| 3. Analizar |
| 4. Reporte |
| 5. Salir |
=====
Ingresa una opción:
```

El menu principal, muestra cualquiera de las 4 opciones. Sin embargo, si se presiona un numero o caracter fuera del esperado, se reiniciará.



1. Cargar Data



Al utilizar la opción 1, se abrirá una ventana en la cual se debe elegir un archivo de entrada de datos

Este archivo solo podrá tener la extensión .data



1. Cargar Data

```
C:/Users/VALLE/Desktop/[LFP] Practica 1 - 202001954 - Copy/LFP_Proyecto1/input.data
ENERO

:741= ( [
"Porvenir (Chile) ",
70.18
,
318
]
;
["Princes Gate (UK)", 176.43,105] ;[ "Recoaro (Italy) ",80.35
,
140 ]; [
"San Pellegrino ", 118.65,3 ]
; ["Santa Barbara (Brazil)"
,
128.97 ,193]; [ "Santa Maria (Mexico) "
, 72.98,194] ;[ "Sao Lourenco (Brazil)",
26.01,87]; [
"Sohat (Lebanon) "
,40.47 ,59] ;[ "Springs (Saudi Arabia)
"
,65.81,308]; [ "Valvert (Belgium)" ,152.54 ,140]; [ "Viladrau (Spain)" ,
;["Vittel (France)",
```

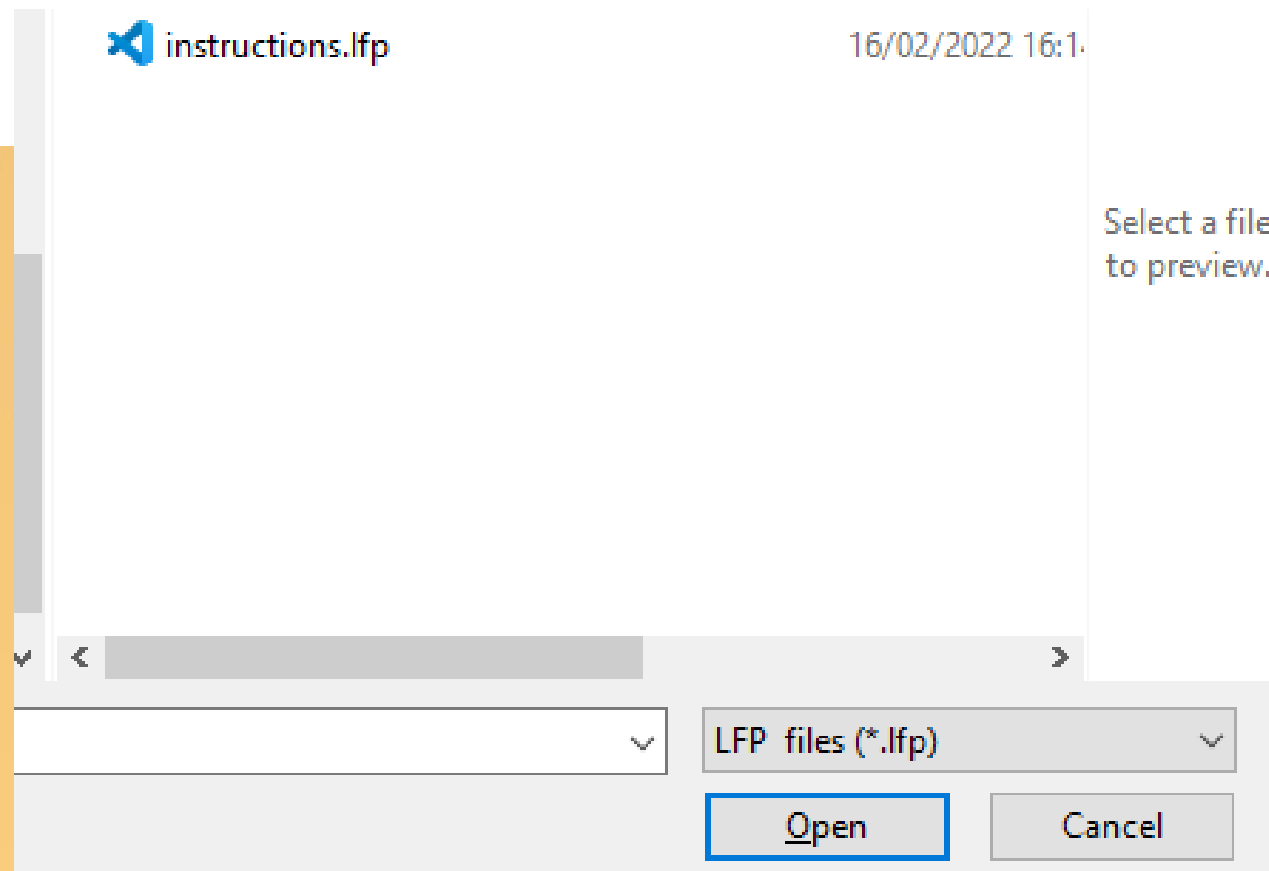
ES importantne recalcar que este archivo es mostrado en pantalla,



```
=====Carga Terminada=====
=====INVENTARIO DE:=====
YEAR -> 741
MONTH -> ENERO
=====Menu =====
| 1. Cargar Data |
| 2. Cargar Instrucciones |
| 3. Analizar |
| 4. Reporte |
| 5. Salir |
=====
Ingresa una opción:
```

Además, también será mostrada la información del reporte

2. Cargar Instrucciones



```
===== Instrucciones cargadas =====  
TITULOY->PRODUCTOY  
TITULOX->PRODUCTOX  
TITULOX->PRODUCTOX  
NOMBRE->PIE  
GRAFICA->PIE  
TITULOY->WHAT  
TITULOX->XD
```








De igual manera, al cargar las instrucciones, la data cargada será mostrada en consola.

Sin embargo, esta data no será la final si es que se escribieron repetidos en el archivo

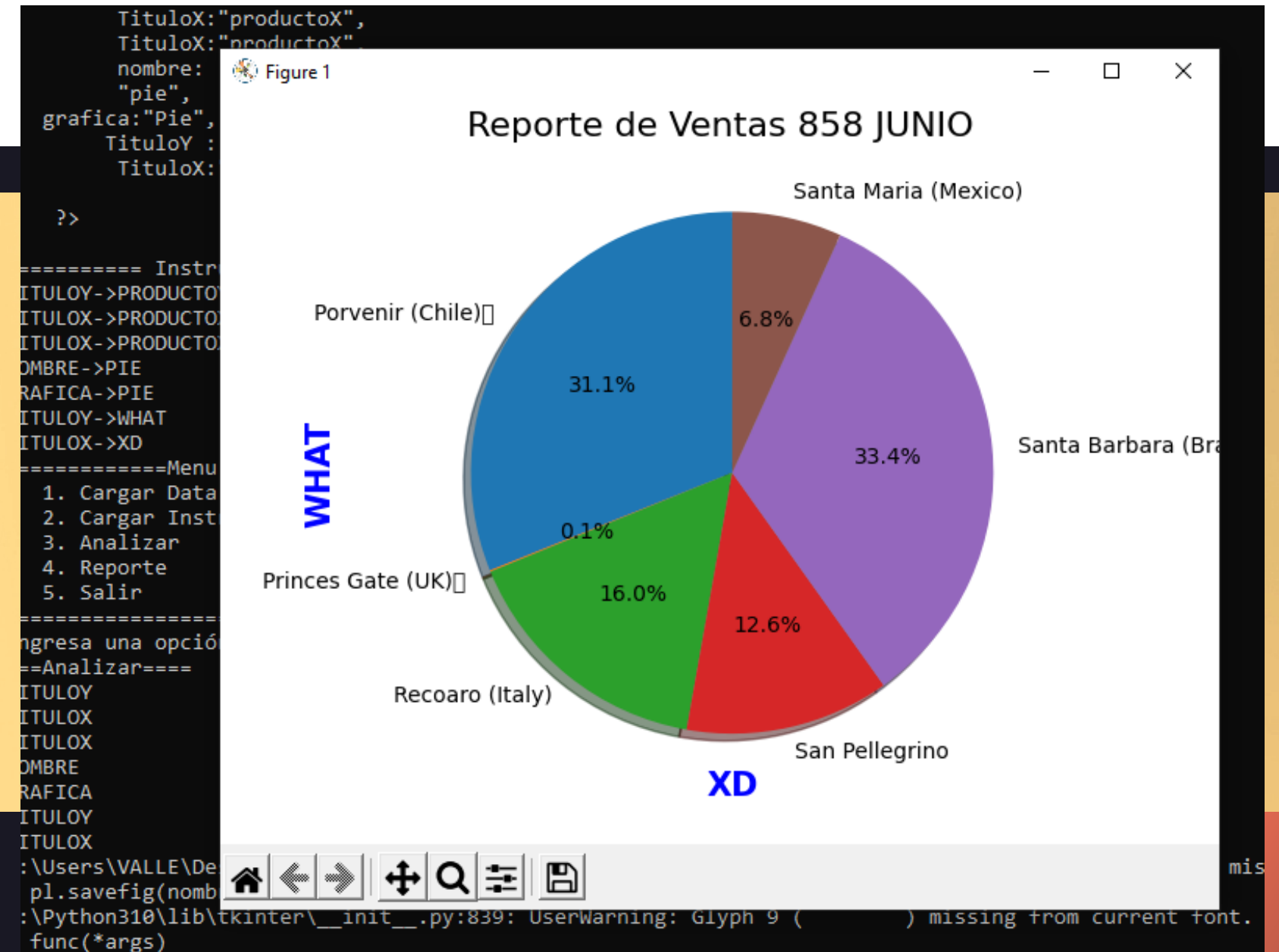


3. ANALIZAR

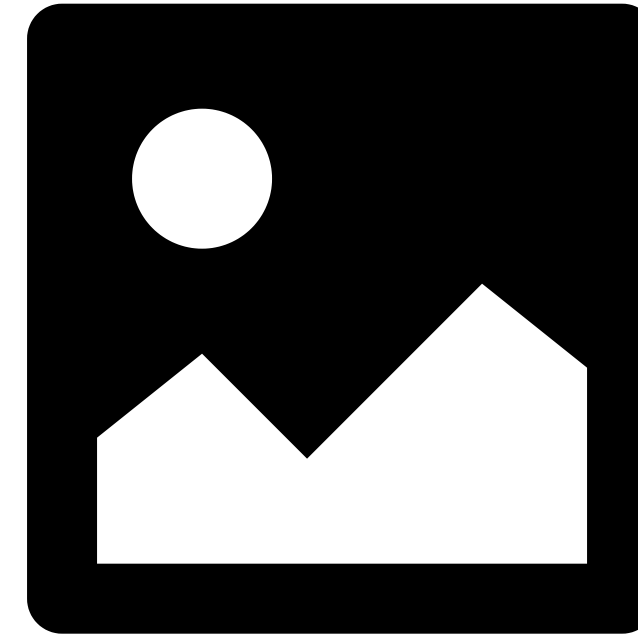
Al presionar analizar, se abrirá una ventana con el respectivo reporte indicado dentro del archivo instrucciones.

	__pycache__	17/02/2022 15:59	File folder	
	models	17/02/2022 15:55	File folder	
	analyzer.py	17/02/2022 15:55	JetBrains PyChar...	10 KB
	app.py	17/02/2022 15:55	JetBrains PyChar...	4 KB
	helpers.py	17/02/2022 15:55	JetBrains PyChar...	8 KB
	instructionsA.py	17/02/2022 15:55	JetBrains PyChar...	7 KB
	PIE.png	17/02/2022 16:10	PNG File	46 KB

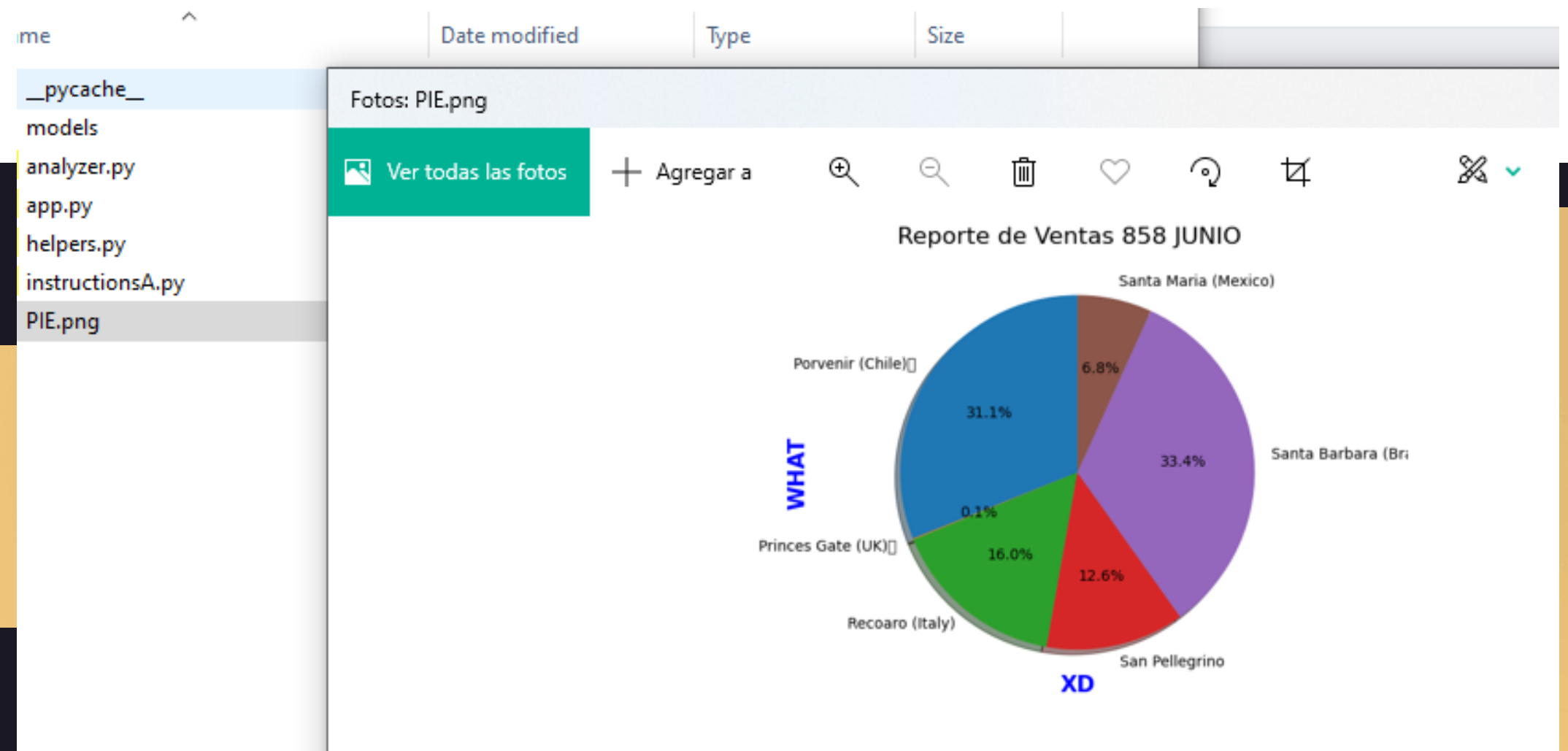
Este será cargado a partir de una imagen png en el archivo del programa



3. ANALIZAR



Esta imagen puede ser abierta

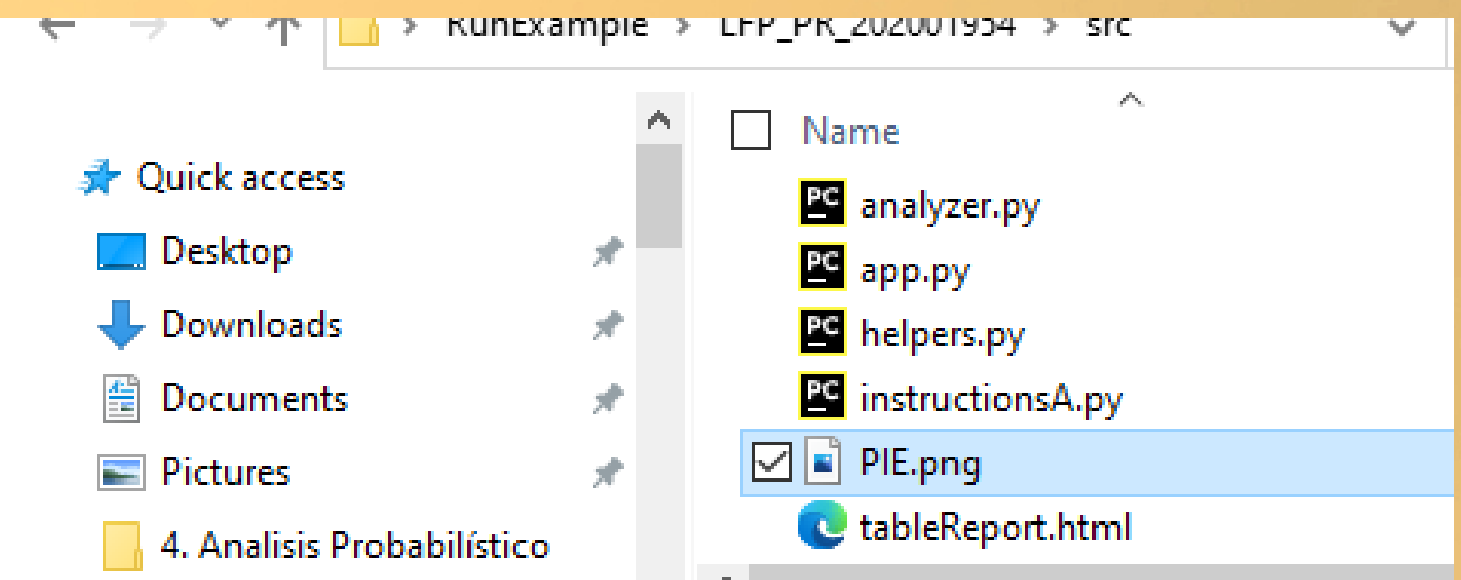


4. REPORTE

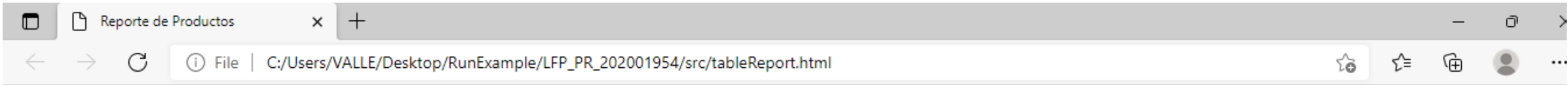
Al presionar analizar, se creará un archivo con el respectivo reporte.

Este archivo tendrá una extensión .html y puede abrirse desde cualquier navegador

```
===Reportes===  
====Se ha generado el reporte====  
=====Menu =====  
| 1. Cargar Data  
| 2. Cargar Instrucciones  
| 3. Analizar  
| 4. Reporte  
| 5. Salir  
=====Ingresar una opción:
```



4. REPORTE



#	Product	Units	Price	Balance
1	Santa Barbara (Brazil)	259	127.22	32949.98
2	Porvenir (Chile)	282	108.91	30712.62
3	Recoaro (Italy)	390	40.52	15802.8
4	San Pellegrino	146	85.5	12483.0
5	Santa Maria (Mexico)	99	67.59	6691.41
6	Princes Gate (UK)	11	8.11	89.21

Productos con Mayores y Menores Ventas				
#	Product	Units	Price	Balance
1	Santa Barbara (Brazil)	259	127.22	32949.98
2	Princes Gate (UK)	11	8.11	89.21



Productos

PRODUCTO MAS VENDIDO
Y MENOS VENDIDO

5. SALIR

```
Ingresa una opción: 5
```

```
=====
| Gracias por usar el Analizador |
=====
```

Fin