

Universidad Autónoma de Tamaulipas

Facultad de Ingeniería Tampico



ASIGNATURA

PROGRAMACION DE INTERFACES Y

PUERTOS

6. Semestre – Grupo “I”

2025 -1

TRABAJO

Programas en Clase y Ejercicios

UNIDAD

2 - DESARROLLO E IMPLEMENTACIÓN DE CONTROLADORES MEDIANTE INTERFACES Y PUERTOS

Docente: Dr. García Ruiz Alejandro H.

Integrante del Equipo	Nivel de Participación
Izaguirre Cortes Emanuel	33.33%
Turrubiates Mejia Gilberto	33.33%
García Salas Yahir Misael	33.33%
Total:	100%



Índice

Índice	1
Repository(s)	2
P00_Intro	2
P01_imagenes_con_PixMap	3
P02_PlantillaProgramas	4
P03_EjemploDial	5
P04_EjemploSpinBox	7
P05_HorizontalSlider	9
P06_VerticalSlider	11
P07_DoubleSpinBox	13
P08_CarruselImagenes	15
P09_SegundoPlano	17
P10_Graficar_Ejemplo1	20
P11_Graficar_Ejemplo2	21
P12_LecturaConcentradoCalifs	22
P13_RadioButton_Ejemplo	23
P14_RadioButton_Ejemplo2	25
P15_RadioButton_Ejemplo3	27
P16_CheckBox_Ejemplo1	29
E01_Cambio de Grados Centígrados Fahrenheit.....	31
E02_Conversor de Horas (En formato de 24horas) a Segundos del día.....	34
E03_Militros a litros.....	37
E04_Metros a Kilómetros	39
E05_Peso a dólar estadounidense.....	42
E06_Area de un Pentágono.....	44
E07_Teorema de Pitágoras	47
E08_Tabla de multiplicar	49



Repositorio(s)

Actividad	Repositorio
Todas	https://github.com/Emanuel-Izaguirre-Cortes-03/Pip_2025_1_eq_4_i/tree/main/Unidad%202

P00_Intro

The screenshot shows the PyCharm IDE interface. On the left is the project tree, which includes a 'Pip_2025_1_eq_4_i' folder containing various UI files (e.g., P00_Intro.ui, P01_Imagenes_con_PixMap.ui) and Python files (e.g., P00_Intro.py, P01_Imagenes_con_PixMap.py). The main editor window displays the code for P00_Intro.py:

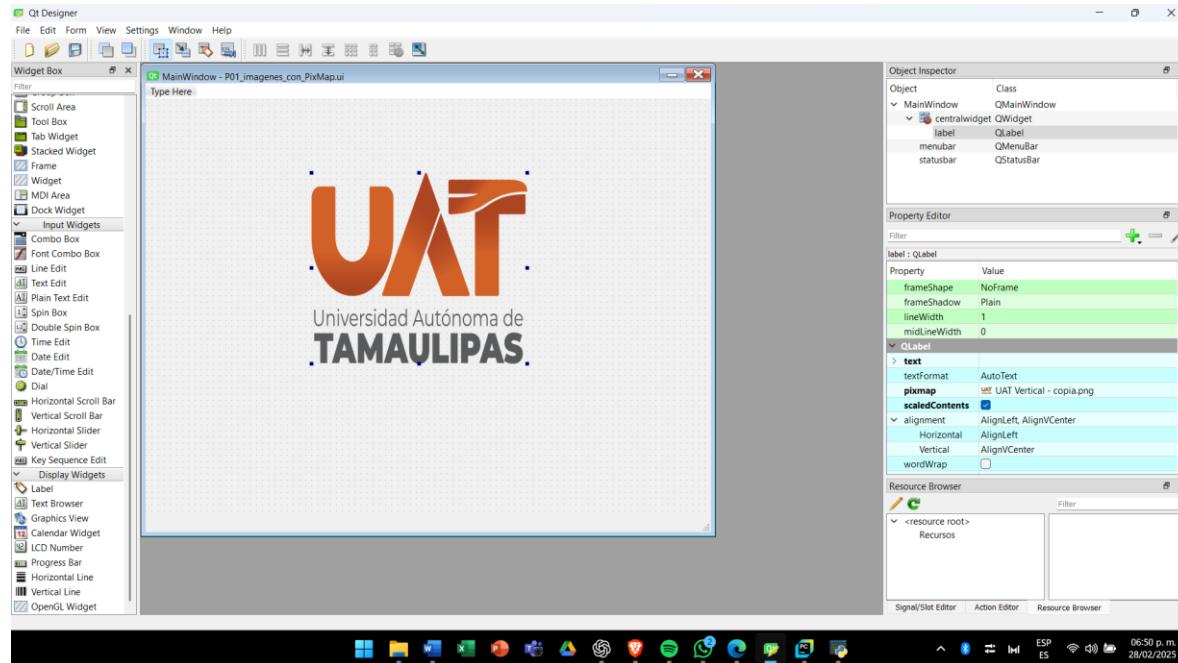
```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 qtCreatorFile = "P00_Intro.ui" # Nombre del archivo aquí.
4 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
6     def __init__(self): # Emanuel-Izaguirre-Cortes-03
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11
12        # Área de los Slots
13    if __name__ == "__main__":
14        app = QtWidgets.QApplication(sys.argv)
15        window = MyApp()
16        window.show()
17        sys.exit(app.exec_())
18
```

To the right of the editor is a small window titled 'MainWindow' showing a blank white space, representing the running application. The status bar at the bottom indicates the current time as 13:27 and date as 28/02/2025.



P01_imagenes_con_PixMap

The screenshot shows the PyCharm IDE interface. On the left, the code editor displays the Python script `P01_imagenes_con_PixMap.py`. The code imports `sys` and `QtWidgets`, creates a UI file `P01_imagenes_con_PixMap.ui`, and defines a `MyApp` class that inherits from `QMainWindow`. The `__init__` method sets up the UI and shows the window. A conditional block checks if the script is run directly (`__name__ == "__main__"`) and creates a QApplication instance, a window object, and shows it. The right side of the screen shows the application window titled "MainWindow" displaying the UAT logo and text.





P02. PlantillaProgramas

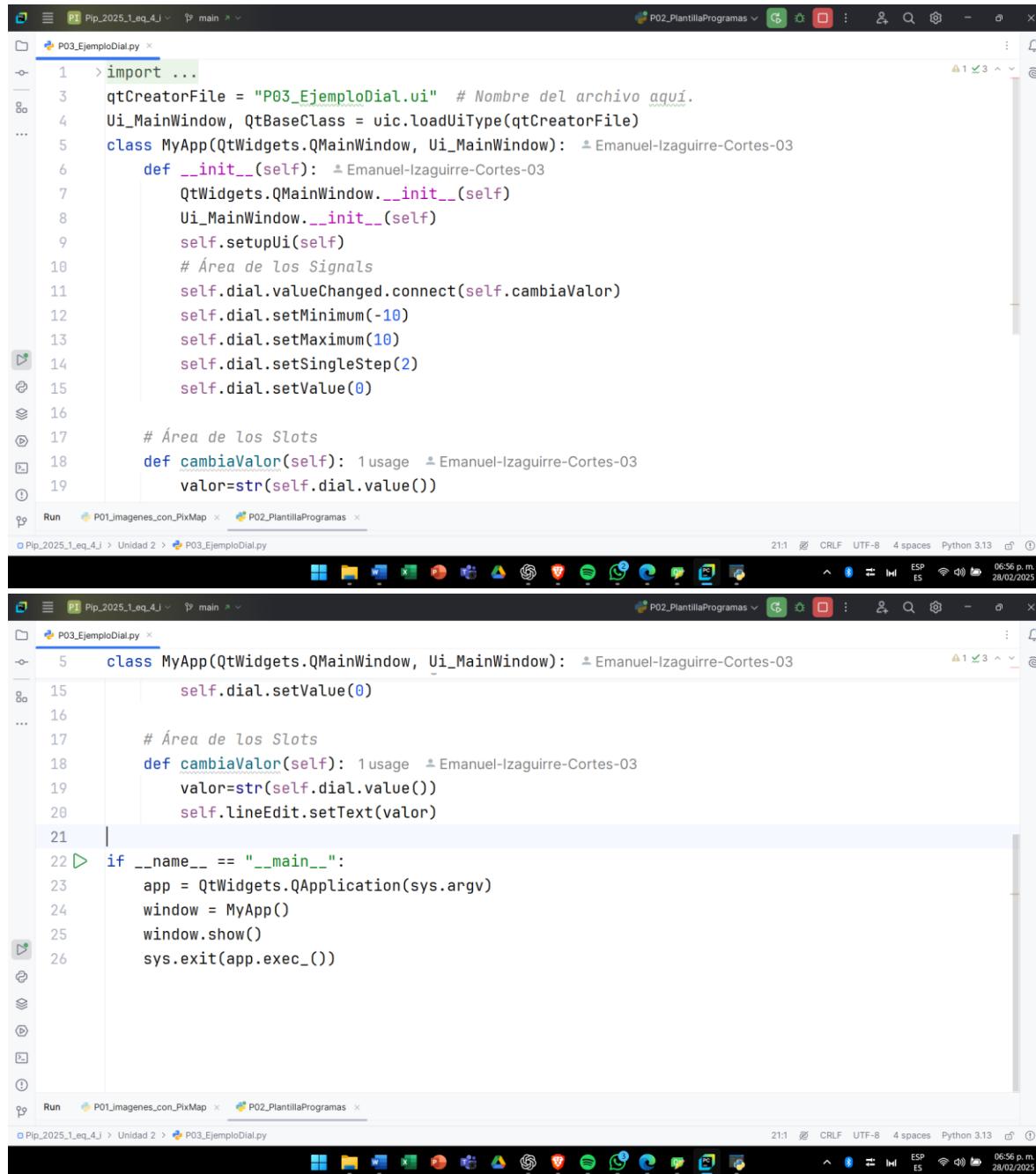


The screenshot shows a Qt application window titled "MainWindow" running in a Windows environment. The window displays the university's logo and name. Below the window, the Qt Designer interface is visible, showing the "Qt Designer" menu bar and various toolbars. A central workspace contains the "MainWindow" window, which is a copy of the one shown above. To the right of the workspace are several floating panels: "Object Inspector", "Property Editor", and "Resource Browser". The "Object Inspector" lists the window's components, including "centralwidget", "label_3", "label_4", "label_5", "menubar", and "statusbar". The "Property Editor" shows properties like "locale", "windowFilePath", "inputMethodHints", "windowModality", "iconSize", "toolButtonStyle", "animated", "documentMode", "tabShape", "dockNestingEnabled", and "dockOptions". The "Resource Browser" shows a single resource entry: "<resource root> Recursos". At the bottom of the screen, the Windows taskbar is visible with various pinned icons.

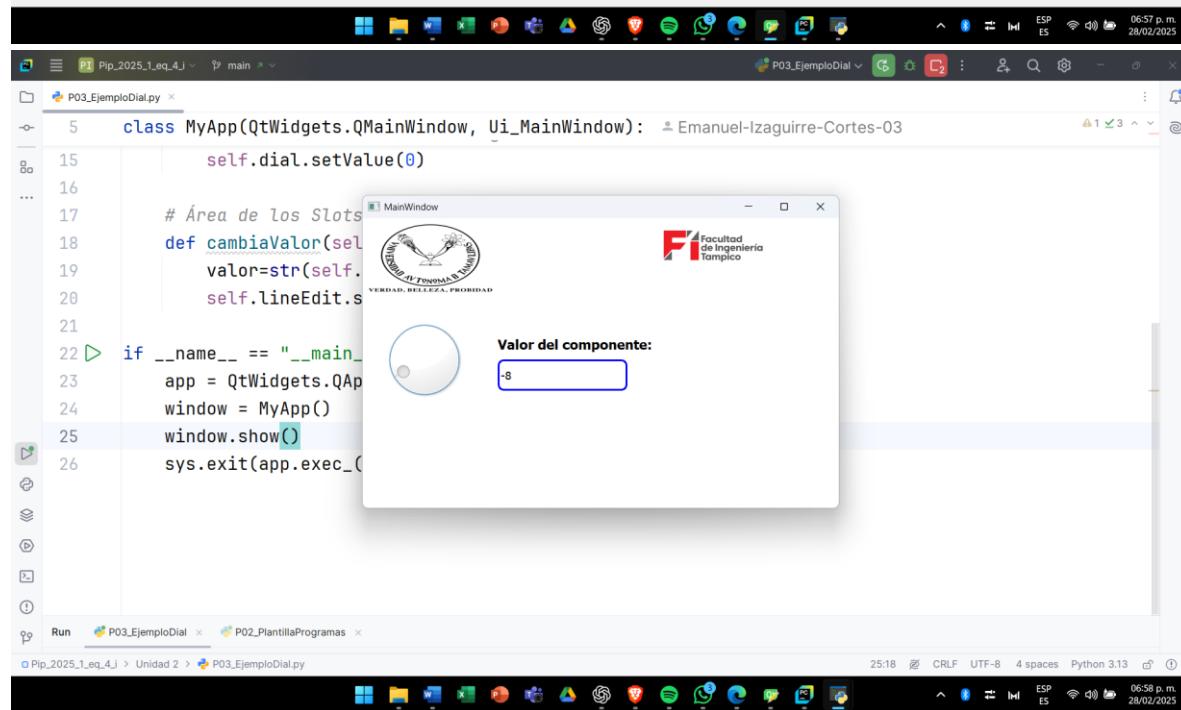
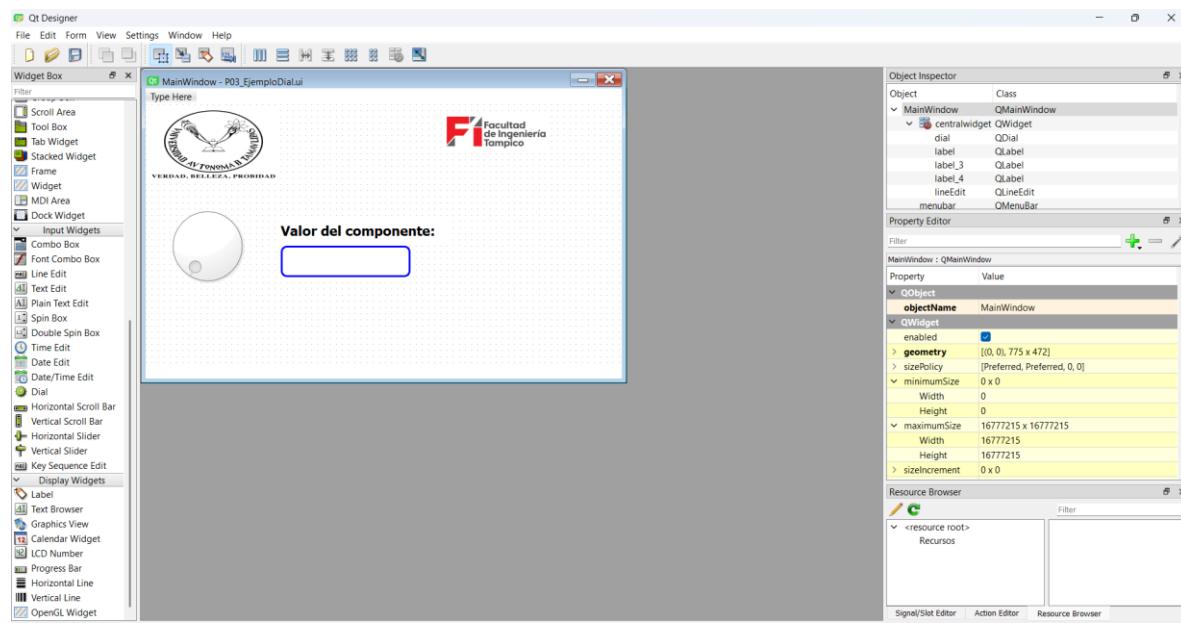
```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 qtCreatorFile = "P02_PlantillaProgramas.ui" # Nombre del archivo aquí.
4 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
6     def __init__(self): # Emanuel-Izaguirre-Cortes-03
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11
12        # Área de los Slots
13    if __name__ == "__main__":
14        app = QtWidgets.QApplication(sys.argv)
15        window = MyApp()
16        window.show()
17        sys.exit(app.exec_())
```



P03_EjemploDial

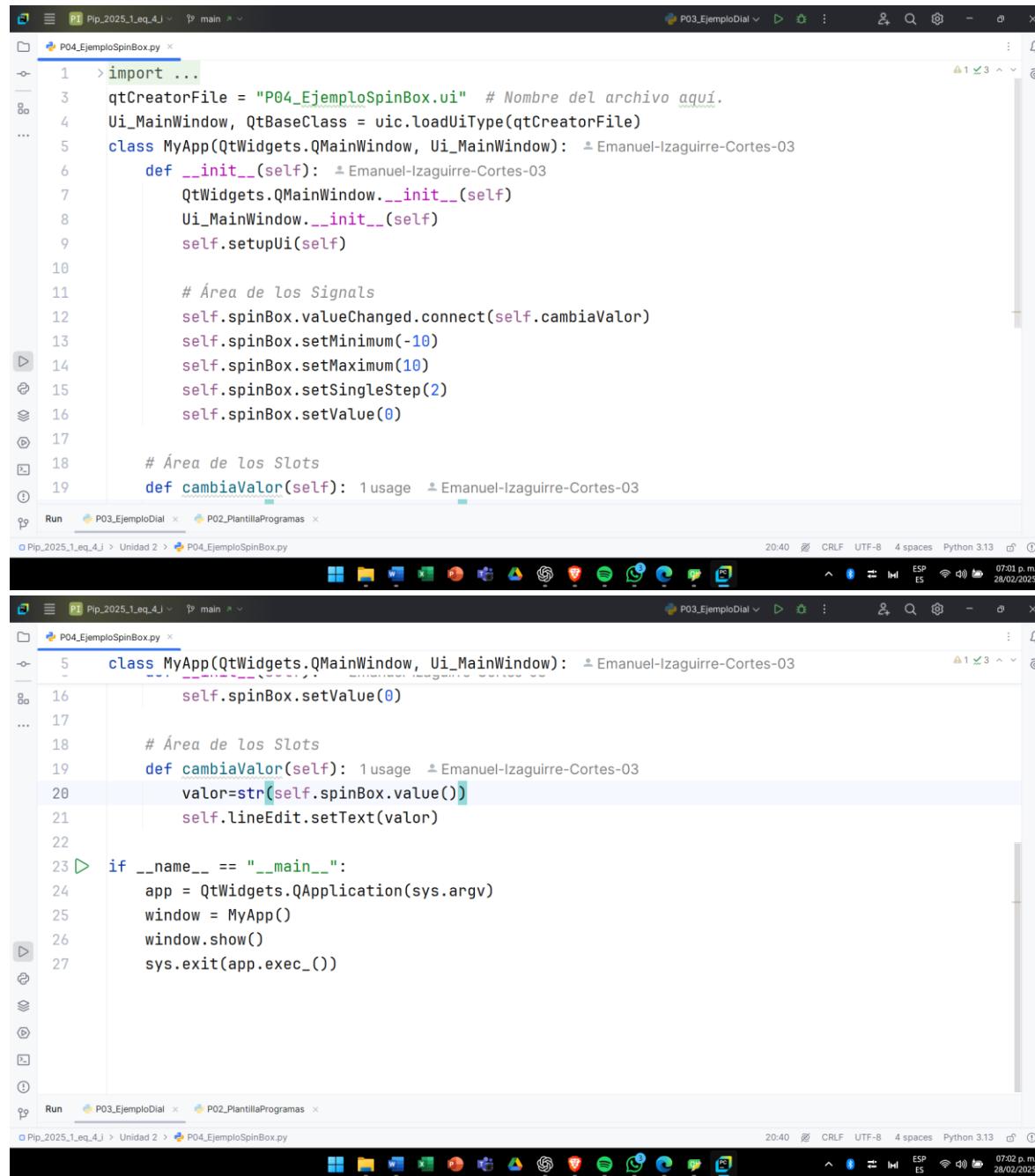


```
1 > import ...
2 qtCreatorFile = "P03_EjemploDial.ui" # Nombre del archivo aquí.
3 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
4
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): __Emanuel-Izaguirre-Cortes-03
6     def __init__(self): __Emanuel-Izaguirre-Cortes-03
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11        self.dial.valueChanged.connect(self.cambiaValor)
12        self.dial.setMinimum(-10)
13        self.dial.setMaximum(10)
14        self.dial.setSingleStep(2)
15        self.dial.setValue(0)
16
17        # Área de los Slots
18    def cambiaValor(self): 1 usage __Emanuel-Izaguirre-Cortes-03
19        valor=str(self.dial.value())
20
21
22 if __name__ == "__main__":
23     app = QtWidgets.QApplication(sys.argv)
24     window = MyApp()
25     window.show()
26     sys.exit(app.exec_())
```

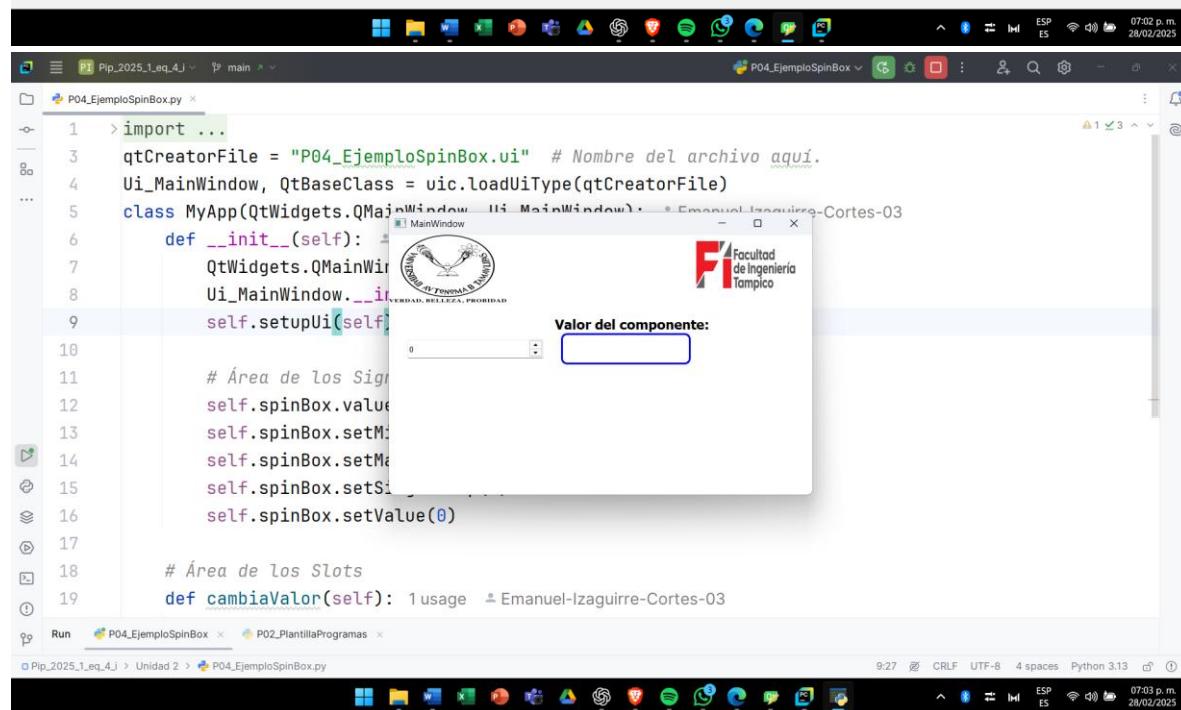
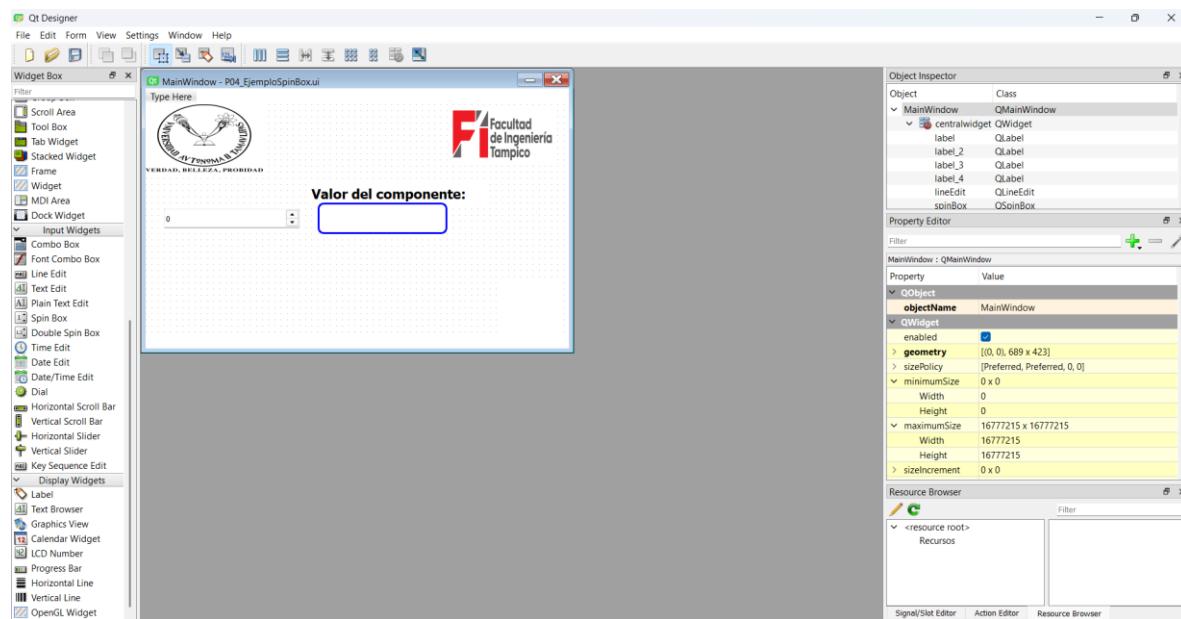




P04_EjemploSpinBox

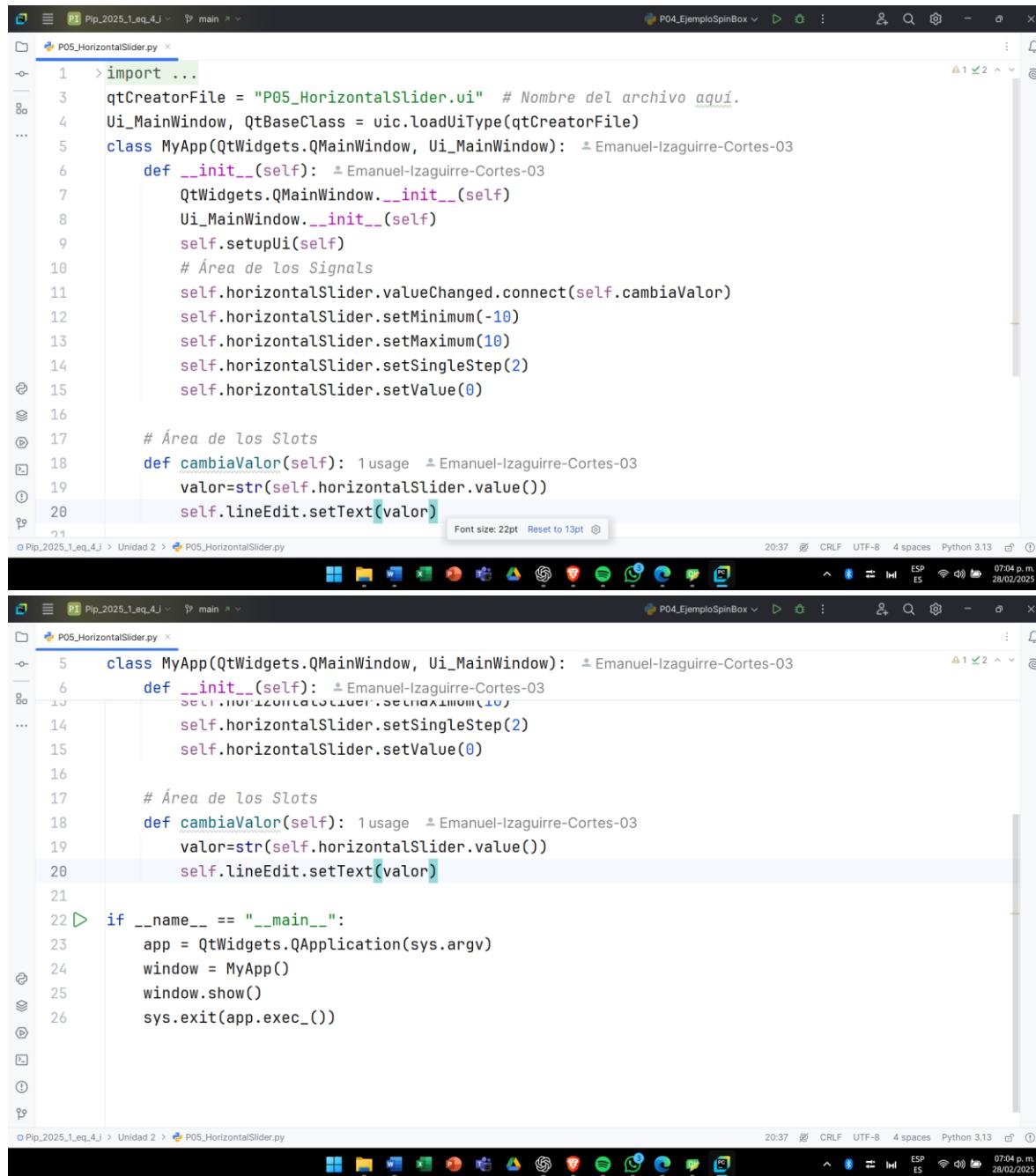


```
1 > import ...
2 qtCreatorFile = "P04_EjemploSpinBox.ui" # Nombre del archivo aquí.
3 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
4
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): __Emanuel-Izaguirre-Cortes-03
6     def __init__(self): __Emanuel-Izaguirre-Cortes-03
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10
11         # Área de los Signals
12         self.spinBox.valueChanged.connect(self.cambiaValor)
13         self.spinBox.setMinimum(-10)
14         self.spinBox.setMaximum(10)
15         self.spinBox.setSingleStep(2)
16         self.spinBox.setValue(0)
17
18         # Área de los Slots
19         def cambiaValor(self): 1 usage __Emanuel-Izaguirre-Cortes-03
20             valor=str(self.spinBox.value())
21             self.lineEdit.setText(valor)
22
23 if __name__ == "__main__":
24     app = QtWidgets.QApplication(sys.argv)
25     window = MyApp()
26     window.show()
27     sys.exit(app.exec_())
```





P05_HorizontalSlider



```
1 > import ...
2 qtCreatorFile = "P05_HorizontalSlider.ui" # Nombre del archivo aquí.
3 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
4
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): __Emanuel-Izaguirre-Cortes-03
6     def __init__(self): __Emanuel-Izaguirre-Cortes-03
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11        self.horizontalSlider.valueChanged.connect(self.cambiaValor)
12        self.horizontalSlider.setMinimum(-10)
13        self.horizontalSlider.setMaximum(10)
14        self.horizontalSlider.setSingleStep(2)
15        self.horizontalSlider.setValue(0)
16
17        # Área de los Slots
18    def cambiaValor(self): 1usage __Emanuel-Izaguirre-Cortes-03
19        valor=str(self.horizontalSlider.value())
20        self.lineEdit.setText(valor)
21
22 if __name__ == "__main__":
23     app = QtWidgets.QApplication(sys.argv)
24     window = MyApp()
25     window.show()
26     sys.exit(app.exec_())
```



Qt Designer

File Edit Form View Settings Window Help

Widget Box

Filter

- Scroll Area
- Tool Box
- Tab Widget
- Stacked Widget
- Frame
- Widget
- MDI Area
- Dock Widget
- Input Widgets
 - Combo Box
 - Font Combo Box
 - Line Edit
 - Text Edit
 - Plain Text Edit
 - Spin Box
 - Double Spin Box
 - Time Edit
 - Date Edit
 - Date/Time Edit
 - Dial
 - Horizontal Scroll Bar
 - Vertical Scroll Bar
 - Horizontal Slider
 - Vertical Slider
- Key Sequence Edit
- Display Widgets
 - Label
 - Text Browser
 - Graphics View
 - Calendar Widget
 - LCD Number
 - Progress Bar
 - Horizontal Line
 - Vertical Line
 - OpenGL Widget

Valor del componente:

4

Pip_2025_1_eq_4J main P05_HorizontalSlider

```
class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("P05_HorizontalSlider")
        self.horizontalSlider.setSingleStep(2)
        self.horizontalSlider.valueChanged.connect(self.cambiaValor)

    def cambiaValor(self):
        valor=str(self.horizontalSlider.value())
        self.lineEdit.setText(valor)

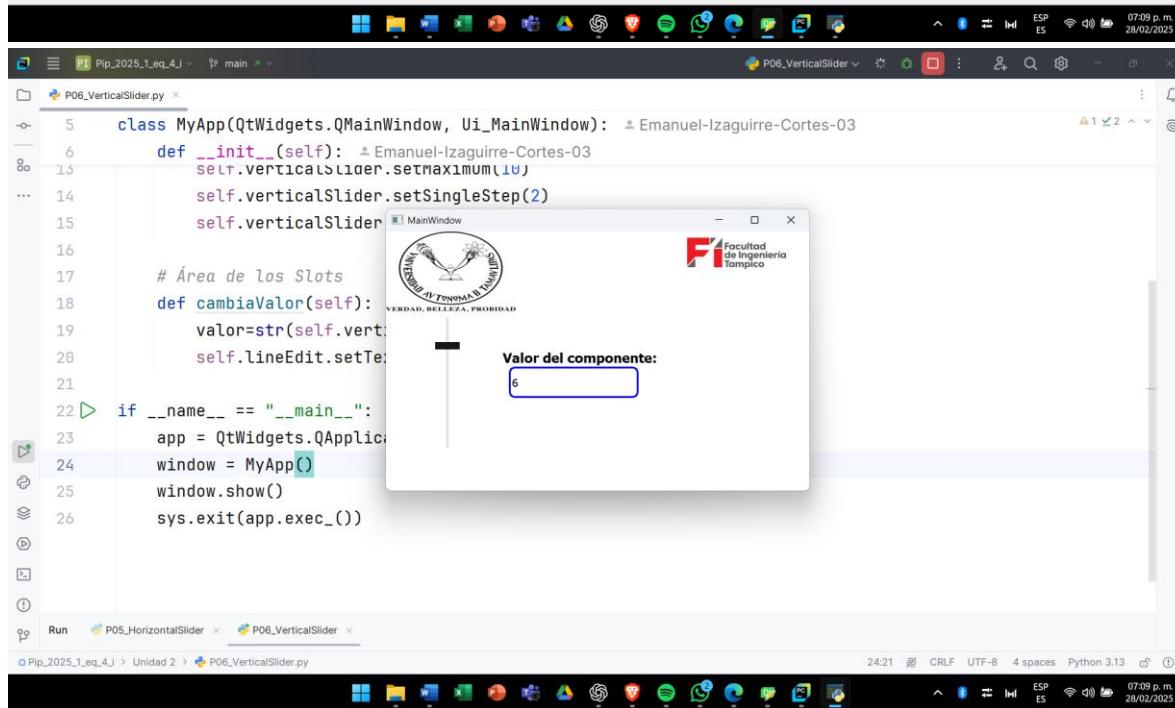
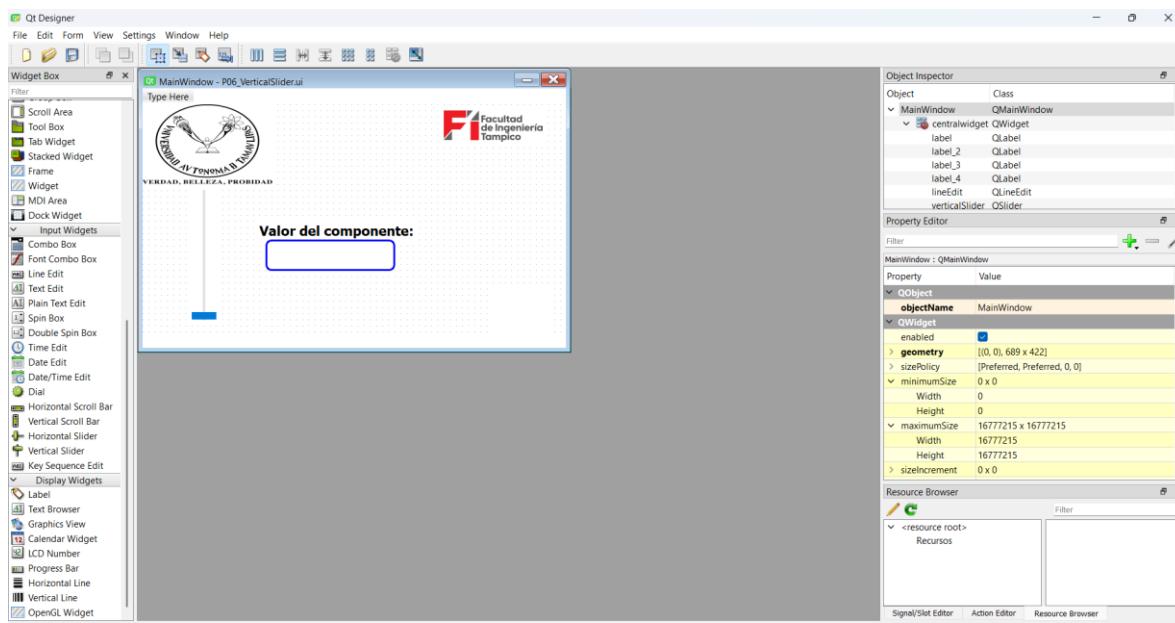
if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    window = MyApp()
    window.show()
    sys.exit(app.exec_())
```



P06_VirtualSlider

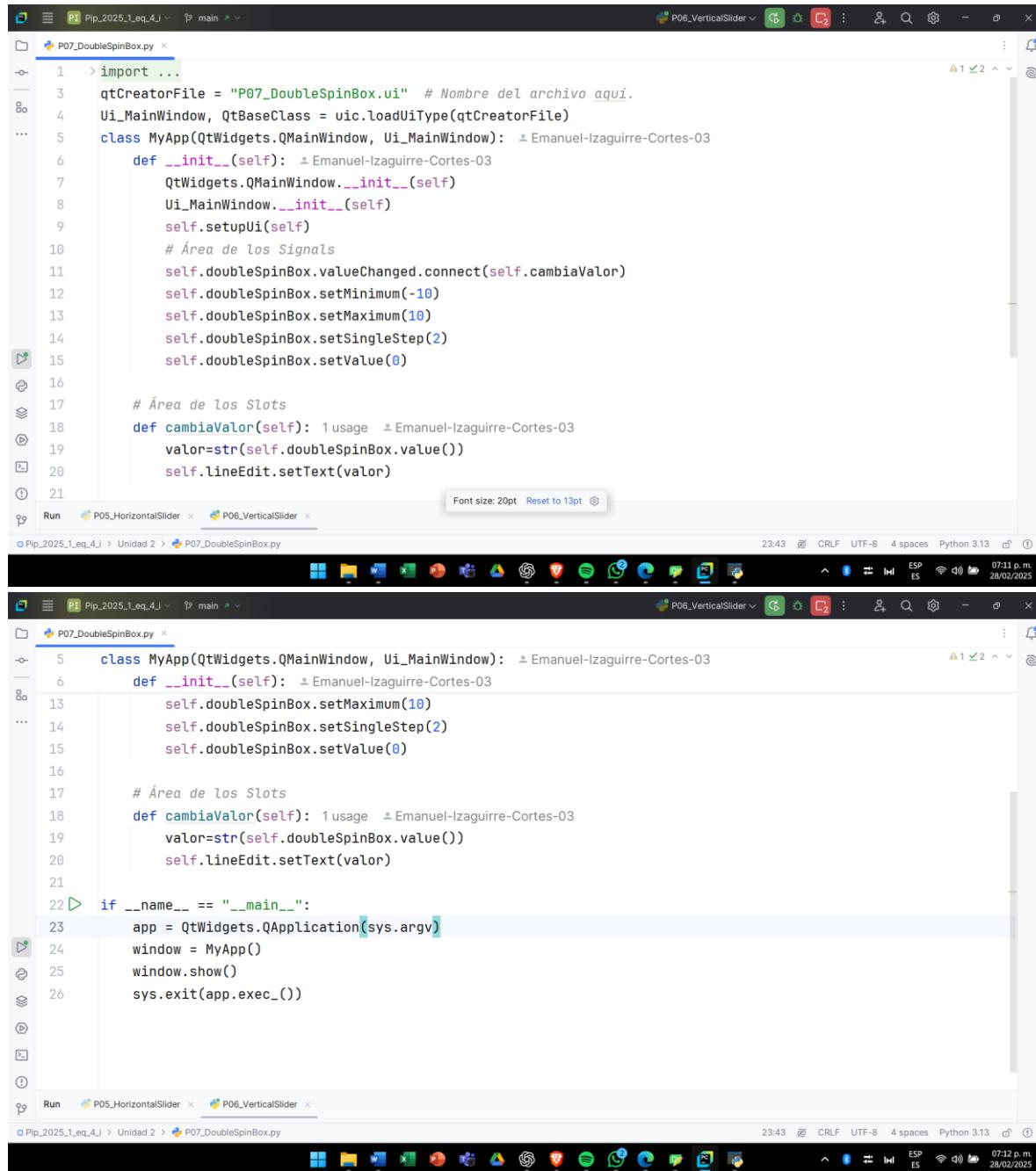
The image shows two side-by-side code editors, both displaying the same Python script named P06_VerticalSlider.py. The script uses PyQt5 to create a window with a vertical slider and a line edit. The code defines a class MyApp that inherits from QMainWindow. It loads a UI file named P06_VerticalSlider.ui and sets up signals and slots. The vertical slider's value is connected to a slot function called cambiaValor, which updates the text of a line edit.

```
1 > import ...
2 qtCreatorFile = "P06_VerticalSlider.ui" # Nombre del archivo aqui.
3 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
4
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
6     def __init__(self):
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11        self.verticalSlider.valueChanged.connect(self.cambiaValor)
12        self.verticalSlider.setMinimum(-10)
13        self.verticalSlider.setMaximum(10)
14        self.verticalSlider.setSingleStep(2)
15        self.verticalSlider.setValue(0)
16
17        # Área de los Slots
18    def cambiaValor(self):
19        valor=str(self.verticalSlider.value())
20        self.lineEdit.setText(valor)
21
22 if __name__ == "__main__":
23     app = QtWidgets.QApplication(sys.argv)
24     window = MyApp()
25     window.show()
26     sys.exit(app.exec_())
```

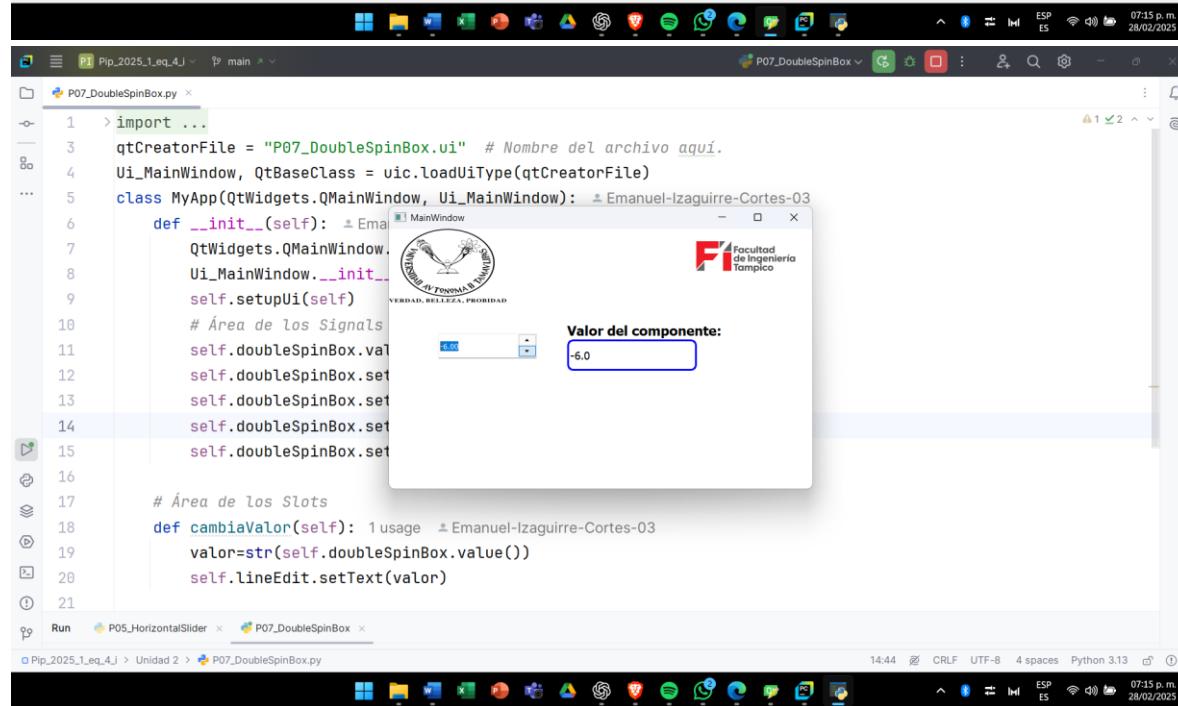
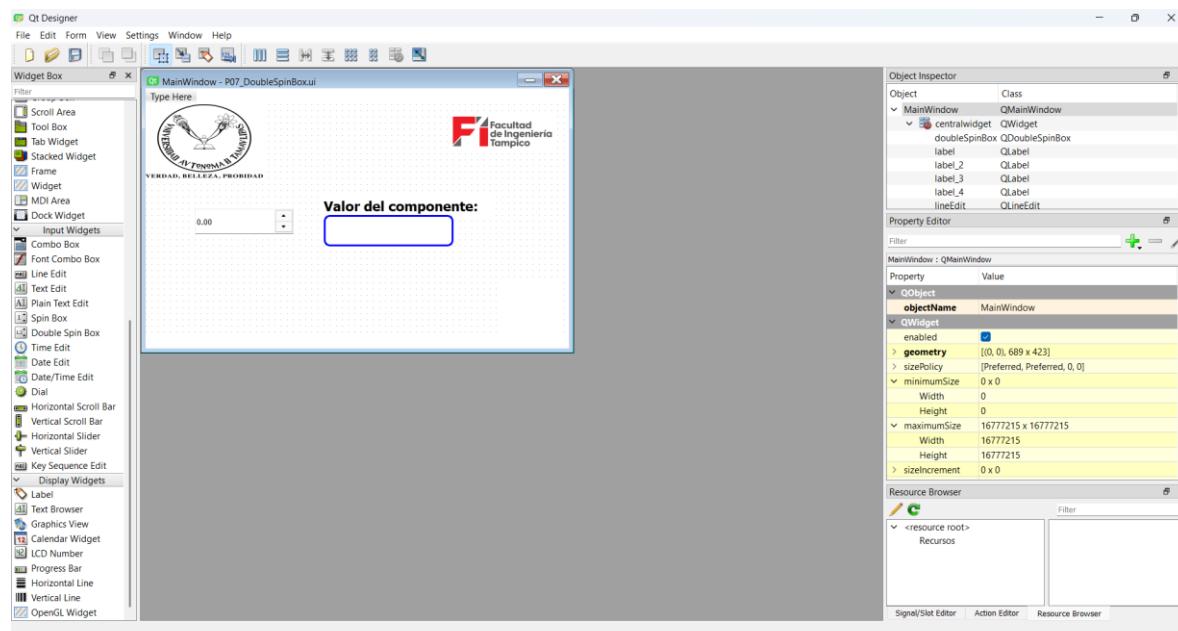




P07_ DoubleSpinBox

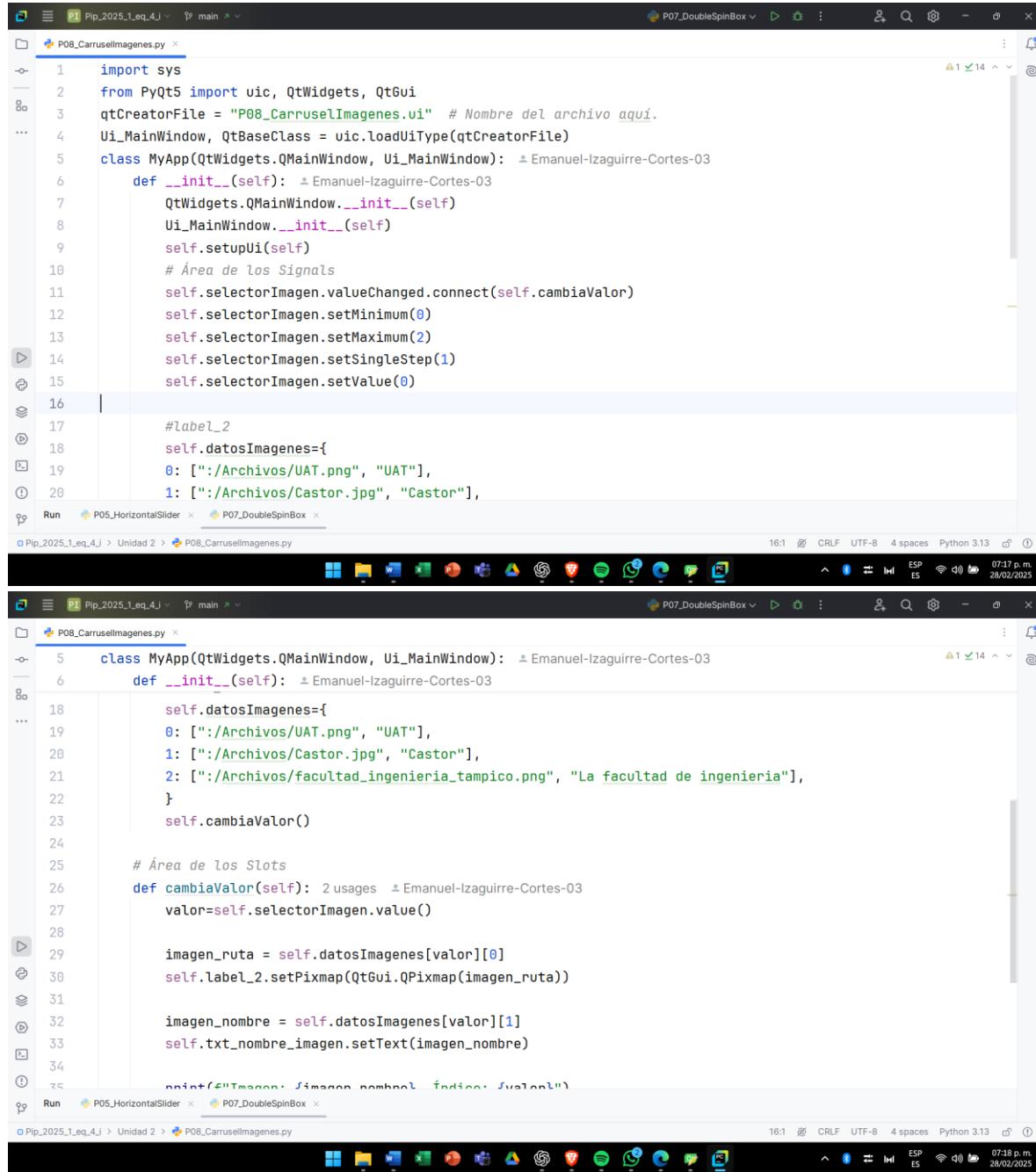


```
1 > import ...
2 qtCreatorFile = "P07_DoubleSpinBox.ui" # Nombre del archivo aquí.
3 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
4
5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
6     def __init__(self):
7         QtWidgets.QMainWindow.__init__(self)
8         Ui_MainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11        self.doubleSpinBox.valueChanged.connect(self.cambiaValor)
12        self.doubleSpinBox.setMinimum(-10)
13        self.doubleSpinBox.setMaximum(10)
14        self.doubleSpinBox.setSingleStep(2)
15        self.doubleSpinBox.setValue(0)
16
17        # Área de los Slots
18    def cambiaValor(self):
19        valor=str(self.doubleSpinBox.value())
20        self.lineEdit.setText(valor)
21
22 if __name__ == "__main__":
23     app = QtWidgets.QApplication(sys.argv)
24     window = MyApp()
25     window.show()
26     sys.exit(app.exec_())
```





P08_CarruselImagenes



```
 1 import sys
 2 from PyQt5 import uic, QtWidgets, QtGui
 3 qtCreatorFile = "P08_CarruselImagenes.ui" # Nombre del archivo aquí.
 4 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
 5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): __Emanuel-Izaguirre-Cortes-03
 6     def __init__(self): __Emanuel-Izaguirre-Cortes-03
 7         QtWidgets.QMainWindow.__init__(self)
 8         Ui_MainWindow.__init__(self)
 9         self.setupUi(self)
10         # Área de los Signals
11         self.selectorImagen.valueChanged.connect(self.cambiaValor)
12         self.selectorImagen.setMinimum(0)
13         self.selectorImagen.setMaximum(2)
14         self.selectorImagen.setSingleStep(1)
15         self.selectorImagen.setValue(0)
16
17         #label_2
18         self.datosImagenes={
19             0: [":/Archivos/UAT.png", "UAT"],
20             1: [":/Archivos/Castor.jpg", "Castor"],
```

```
 5 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): __Emanuel-Izaguirre-Cortes-03
 6     def __init__(self): __Emanuel-Izaguirre-Cortes-03
 7
 8         self.datosImagenes{
 9             0: [":/Archivos/UAT.png", "UAT"],
10             1: [":/Archivos/Castor.jpg", "Castor"],
11             2: [":/Archivos/facultad_ingenieria_tampico.png", "La facultad de ingeniería"],
12         }
13         self.cambiaValor()
14
15         # Área de los Slots
16         def cambiaValor(self): 2 usages __Emanuel-Izaguirre-Cortes-03
17             valor=self.selectorImagen.value()
18
19             imagen_ruta = self.datosImagenes[valor][0]
20             self.label_2.setPixmap(QtGui.QPixmap(imagen_ruta))
21
22             imagen_nombre = self.datosImagenes[valor][1]
23             self.txt_nombre_imagen.setText(imagen_nombre)
24
25             print(f"Imagen: {imagen_nombre} - Índice: {valor}")
26
27
28
29
30
31
32
33
34
```



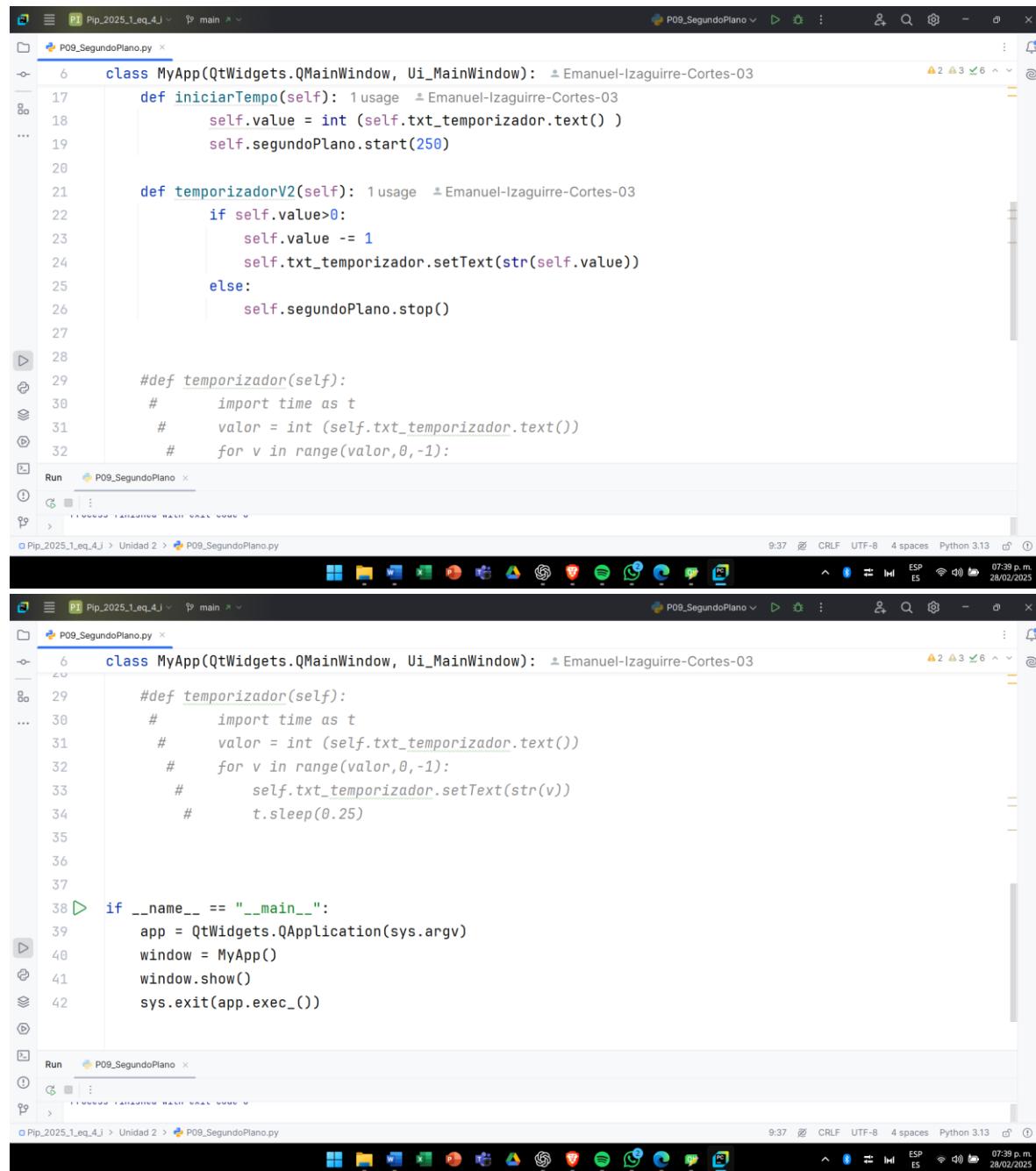
```
5     class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):  Emanuel-Izaguirre-Cortes-03
6         def cambiaValor(self):  2 usages Emanuel-Izaguirre-Cortes-03
7             imagen_ruta = self.datosImagenes[valor][0]
8             self.label_2.setPixmap(QtGui.QPixmap(imagen_ruta))
9
10            imagen_nombre = self.datosImagenes[valor][1]
11            self.txt_nombre_imagen.setText(imagen_nombre)
12
13            print(f"Imagen: {imagen_nombre}, índice: {valor}")
14
15        if __name__ == "__main__":
16            app = QtWidgets.QApplication(sys.argv)
17            window = MyApp()
18            window.show()
19            sys.exit(app.exec_())
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
```



The screenshot shows the PyCharm IDE interface. The code editor displays the Python script `P08_CarruselImagenes.py`. The script defines a class `MyApp` that inherits from `QtWidgets.QMainWindow` and `Ui_MainWindow`. It includes a method `cambiaValor` that prints the current name and changes the image in the central label. A variable `Imagen_nombre` is set to "UAT". A message box is shown with the text "Nombre:" and a text input field containing "UAT". The status bar at the bottom right shows the date and time as 07:33 p.m. 28/02/2025.

P09_SegundoPlano

The screenshot shows the PyCharm IDE interface. The code editor displays the Python script `P09_SegundoPlano.py`. The script imports `sys`, `Recursos_rc`, and `PyQt5`. It uses `QtWidgets`, `QtGui`, and `QtCore`. The `qtCreatorFile` variable is set to "P09_SegundoPlano.ui". The `Ui_MainWindow` class is loaded using `loadUiType`. The `MyApp` class inherits from `QtWidgets.QMainWindow` and `Ui_MainWindow`. It overrides the `__init__` method to call the `__init__` methods of both classes and set up the UI. It also connects the button's click signal to the `iniciarTempo` slot and creates a timer object. The `iniciarTempo` method is defined but not fully implemented. The status bar at the bottom right shows the date and time as 07:39 p.m. 28/02/2025.



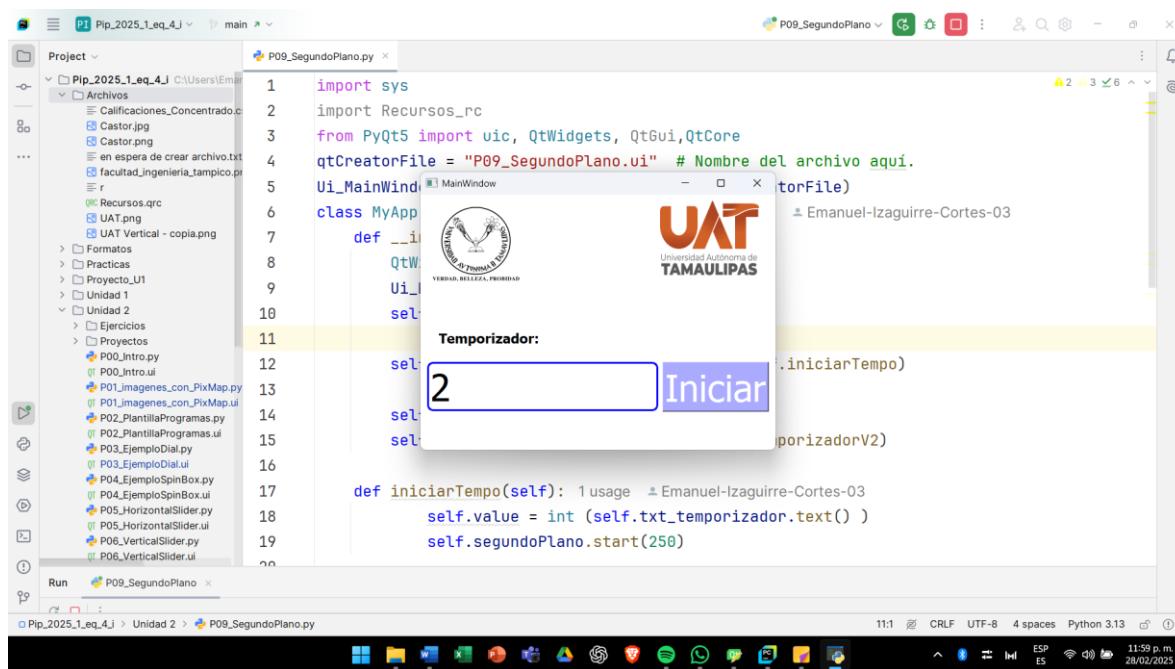
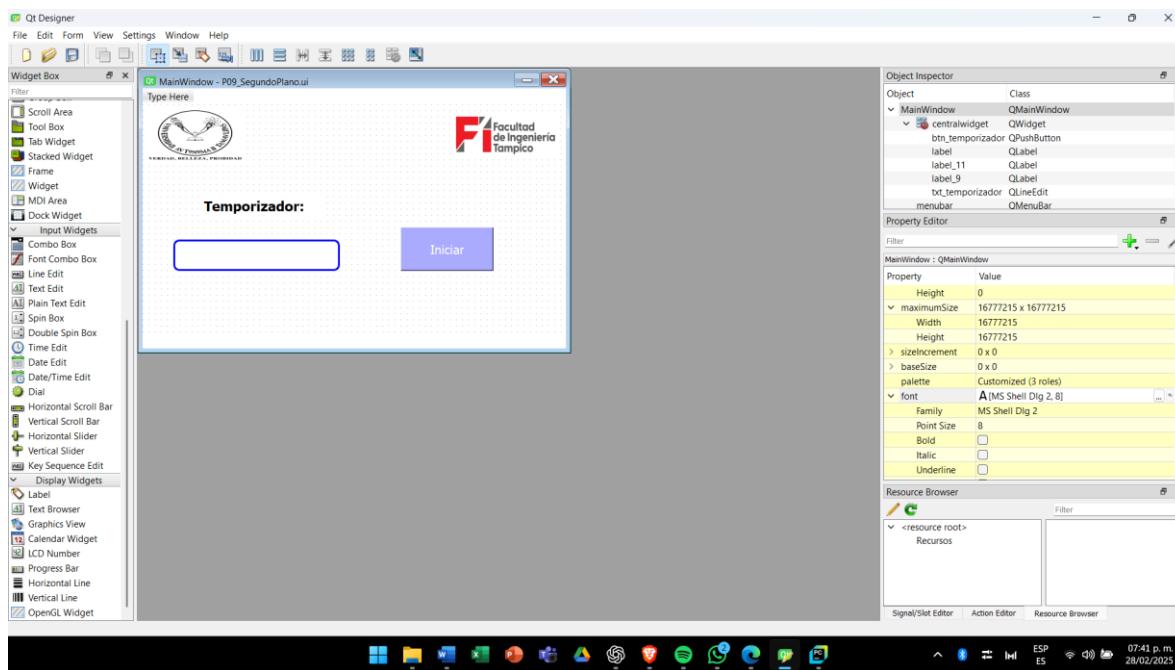
```
Pip_2025_1_eq_4.j main P09_SegundoPlano

6  class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): Emanuel-Izaguirre-Cortes-03
17      def iniciarTempo(self): 1 usage Emanuel-Izaguirre-Cortes-03
18          self.value = int (self.txt_temporizador.text())
19          self.segundoPlano.start(250)
20
21      def temporizadorV2(self): 1 usage Emanuel-Izaguirre-Cortes-03
22          if self.value>0:
23              self.value -= 1
24              self.txt_temporizador.setText(str(self.value))
25          else:
26              self.segundoPlano.stop()
27
28
29      #def temporizador(self):
30      #    import time as t
31      #    valor = int (self.txt_temporizador.text())
32      #    for v in range(valor,0,-1):
33      #        self.txt_temporizador.setText(str(v))
34      #        t.sleep(0.25)
35
36
37
38 if __name__ == "__main__":
39     app = QtWidgets.QApplication(sys.argv)
40     window = MyApp()
41     window.show()
42     sys.exit(app.exec_())

Run P09_SegundoPlano x
Pip_2025_1_eq_4.j Unidad 2 P09_SegundoPlano.py
9:37 CRLF UTF-8 4 spaces Python 3.13 07:39 p.m. 28/02/2025

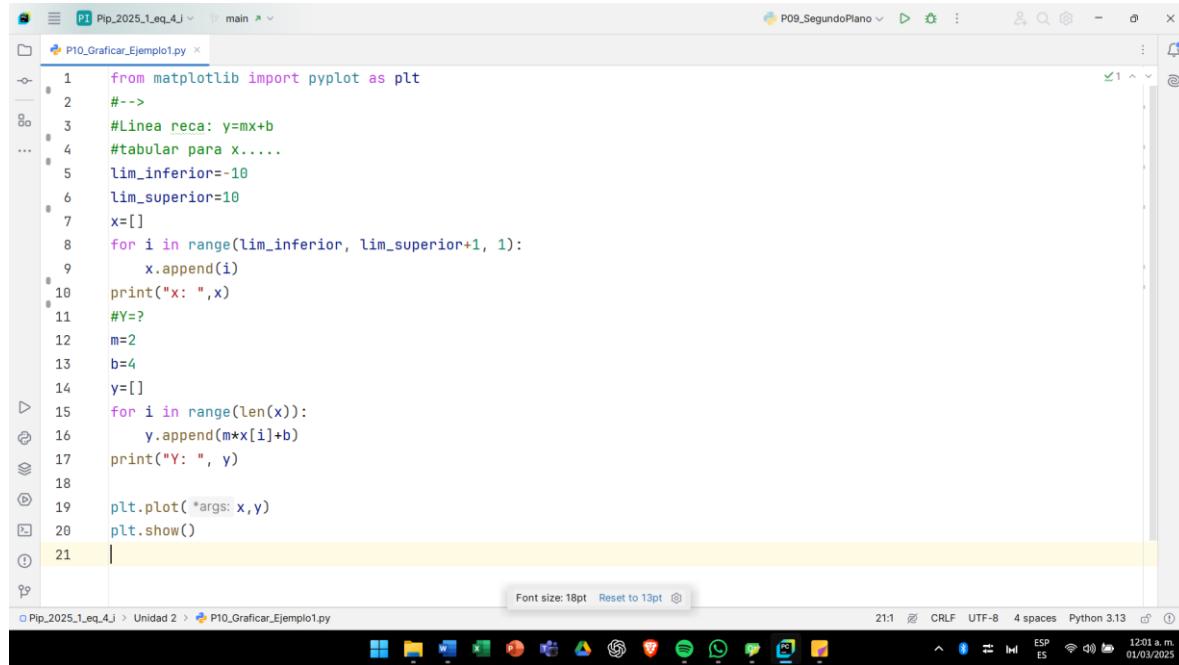
Pip_2025_1_eq_4.j main P09_SegundoPlano
6  class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): Emanuel-Izaguirre-Cortes-03
29      #def temporizador(self):
30      #    import time as t
31      #    valor = int (self.txt_temporizador.text())
32      #    for v in range(valor,0,-1):
33      #        self.txt_temporizador.setText(str(v))
34      #        t.sleep(0.25)
35
36
37
38 if __name__ == "__main__":
39     app = QtWidgets.QApplication(sys.argv)
40     window = MyApp()
41     window.show()
42     sys.exit(app.exec_())

Run P09_SegundoPlano x
Pip_2025_1_eq_4.j Unidad 2 P09_SegundoPlano.py
9:37 CRLF UTF-8 4 spaces Python 3.13 07:39 p.m. 28/02/2025
```

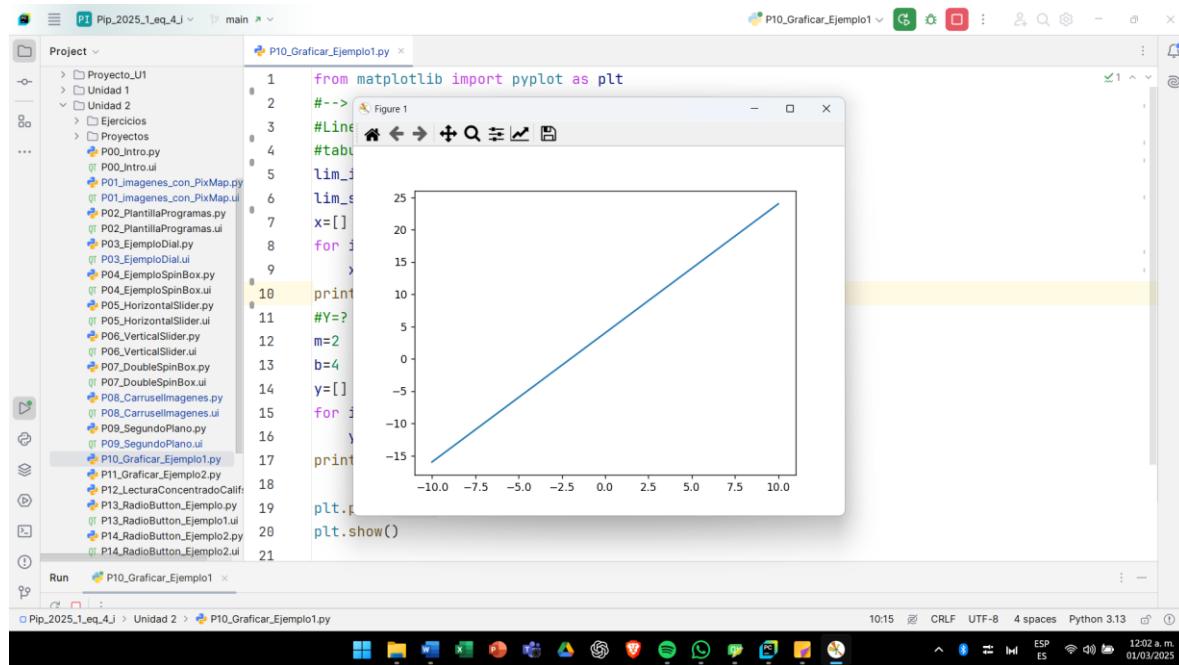




P10_Graficar_Ejemplo1

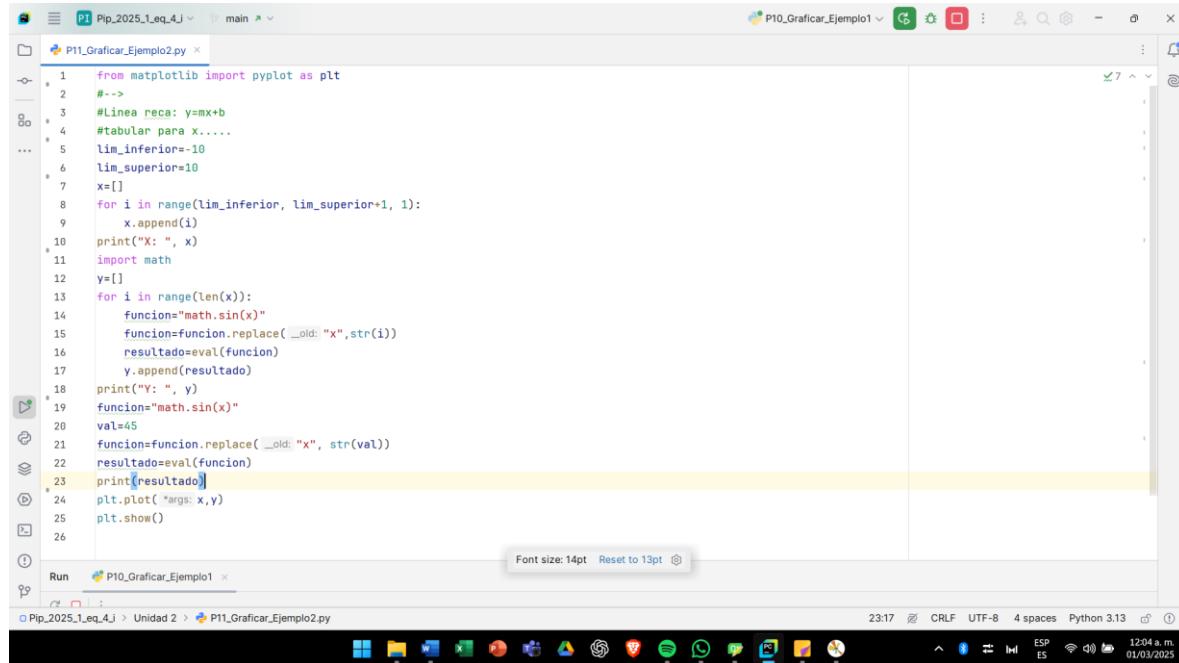


```
from matplotlib import pyplot as plt
#-->
#Linea reca: y=mx+b
#tabular para x.....
lim_inferior=-10
lim_superior=10
x=[]
for i in range(lim_inferior, lim_superior+1, 1):
    x.append(i)
print("x: ",x)
#Y=?
m=2
b=4
y=[]
for i in range(len(x)):
    y.append(m*x[i]+b)
print("Y: ", y)
plt.plot(*args: x,y)
plt.show()
```

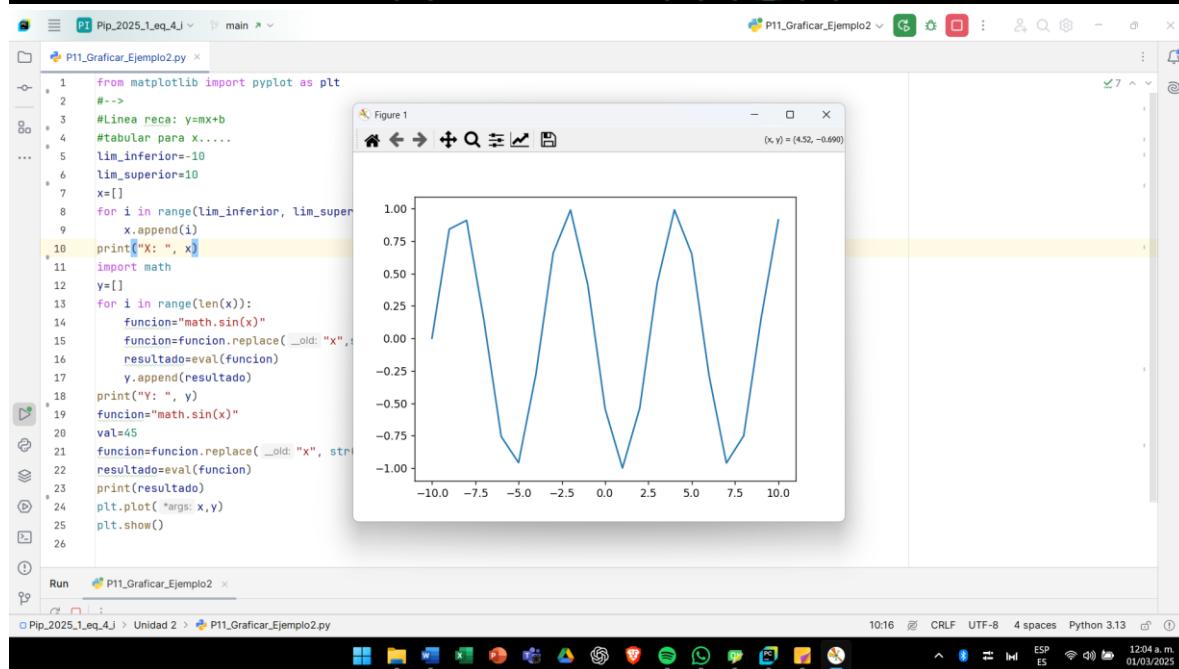




P11_Graficar_Ejemplo2



```
1 from matplotlib import pyplot as plt
2 #-->
3 #Linea reca: y=mx+b
4 #tabular para x.....
5 lim_inferior=-10
6 lim_superior=10
7 x=[]
8 for i in range(lim_inferior, lim_superior+1, 1):
9     x.append(i)
10    print("X: ", x)
11    import math
12    y=[]
13    for i in range(len(x)):
14        funcion="math.sin(x)"
15        funcion=funcion.replace( _old: "x",str(i))
16        resultado=eval(funcion)
17        y.append(resultado)
18    print("Y: ", y)
19    funcion="math.sin(x)"
20    val<45
21    funcion=funcion.replace( _old: "x", str(val))
22    resultado=eval(funcion)
23    print(resultado)
24    plt.plot(*args: x,y)
25    plt.show()
```





P12_LecturaConcentradoCalifs

```
from matplotlib import pyplot as plt
archivo=open("../Archivos/Calificaciones_Concentrado.csv")
contenido=archivo.readlines()
print(contenido)

datos=[]
for linea in contenido:
    datos.append(linea.split(","))
print(datos)

datosT=[]
for registro in datos:
    temporal=[registro[0], list(map(int,registro[1:]))]
    datosT.append(temporal)
print(datosT)

for registro in datosT:
    registro.append(sum(registro[1])/len(registro[1]))
print(datosT)
```

```
print(datos)

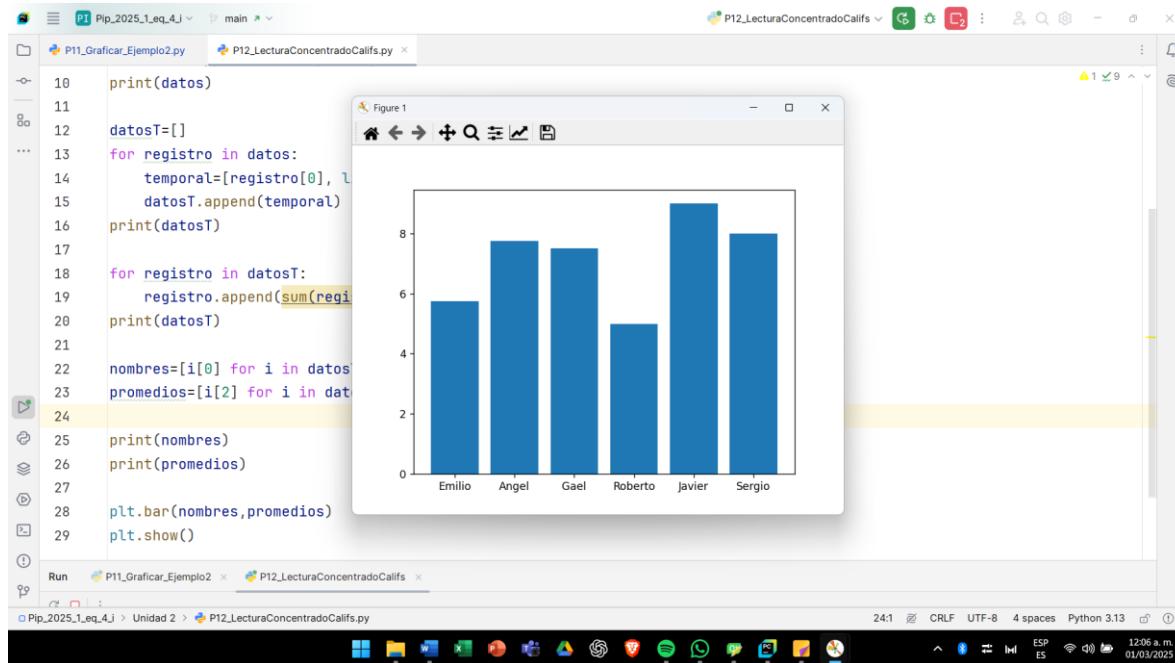
datosT=[]
for registro in datos:
    temporal=[registro[0], list(map(int,registro[1:]))]
    datosT.append(temporal)
print(datosT)

for registro in datosT:
    registro.append(sum(registro[1])/len(registro[1]))
print(datosT)

nombres=[i[0] for i in datosT]
promedios=[i[2] for i in datosT]

print(nombres)
print(promedios)

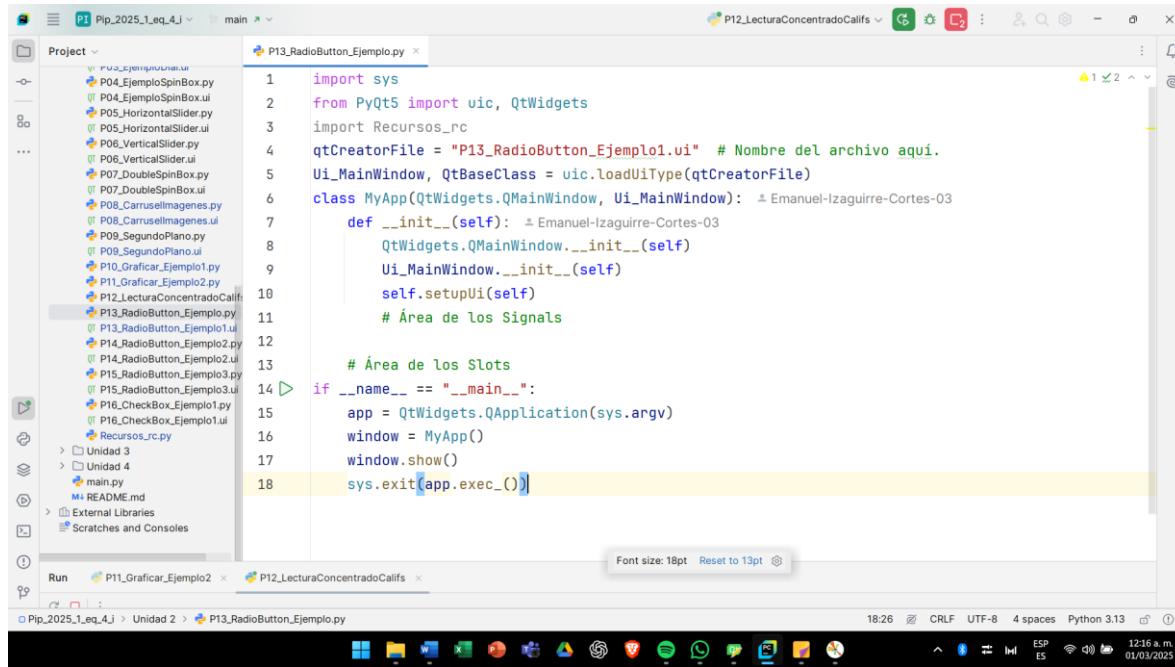
plt.bar(nombres,promedios)
plt.show()
```



A screenshot of the PyCharm IDE. On the left, the code editor shows a Python script named P11_Graficar_Ejemplo2.py. The script contains code to read data from a list, calculate averages, and plot them as a bar chart. On the right, a figure window titled 'Figure 1' displays a bar chart with six bars representing different names: Emilio, Angel, Gael, Roberto, Javier, and Sergio. The y-axis ranges from 0 to 8. The data points are approximately: Emilio (~6), Angel (~7.8), Gael (~7.5), Roberto (~5.2), Javier (~8.5), and Sergio (~7.8). Below the figure window, the status bar shows the date and time as 12:06 a.m. 01/03/2025.

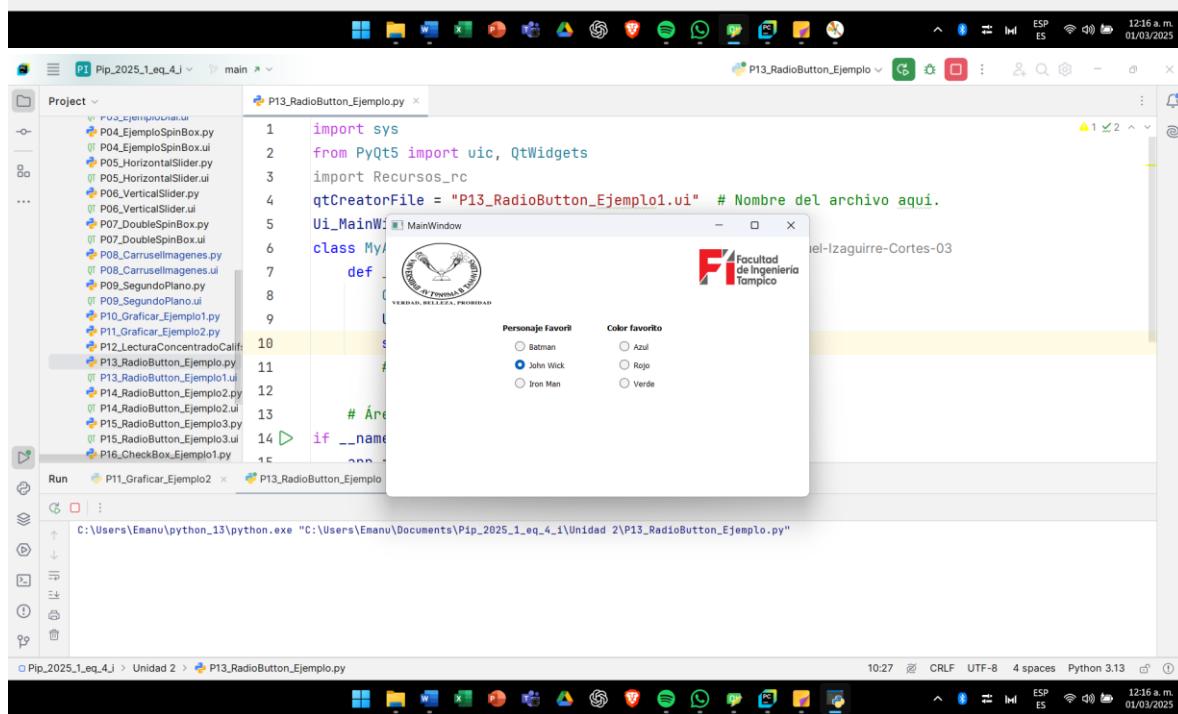
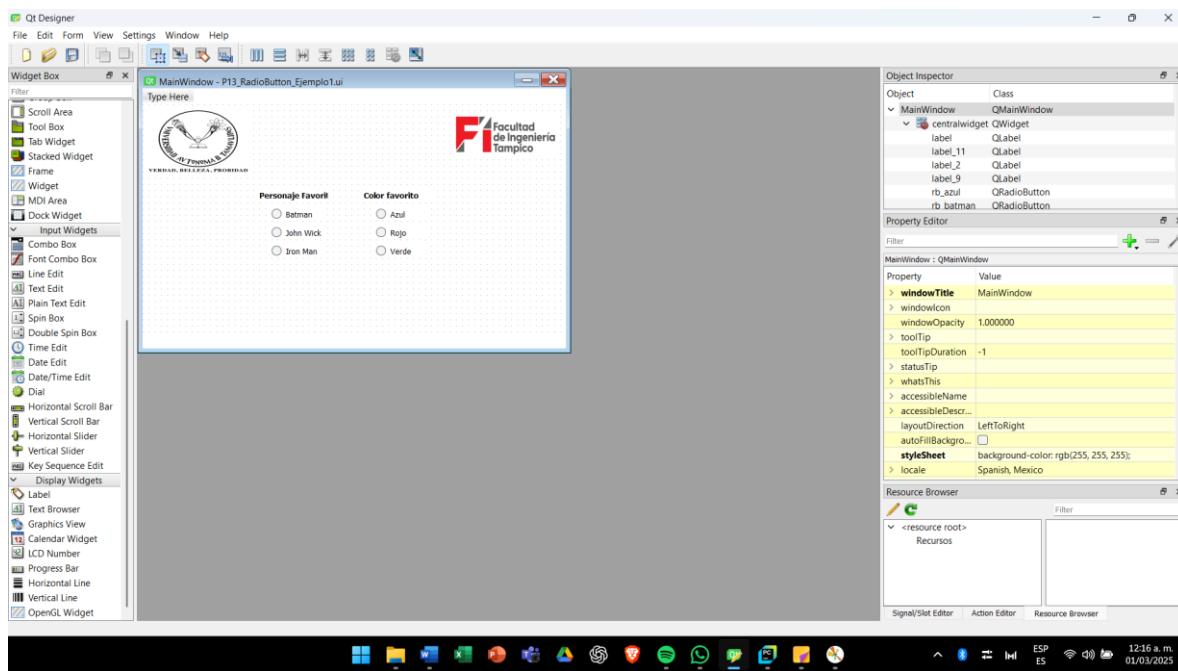
```
10 print(datos)
11
12 datosT=[]
13 for registro in datos:
14     temporal=[registro[0], L
15     datosT.append(temporal)
16 print(datosT)
17
18 for registro in datosT:
19     registro.append(sum(regi
20 print(datosT)
21
22 nombres=[i[0] for i in datos
23 promedios=[i[2] for i in dat
24
25 print(nombres)
26 print(promedios)
27
28 plt.bar(nombres,promedios)
29 plt.show()
```

P13_RadioButton_Ejemplo



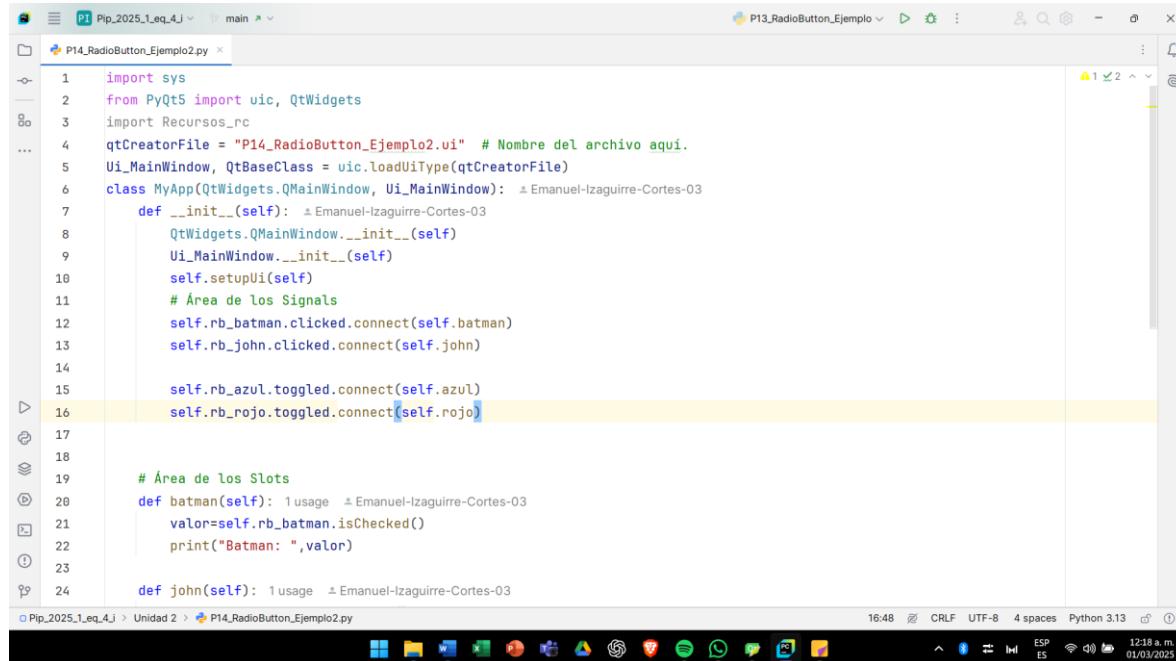
A screenshot of the PyCharm IDE. On the left, the project structure shows multiple .ui files for various examples like SpinBox, Slider, and RadioButton. The main code editor window shows a script named P13_RadioButton_Ejemplo.py. The code uses PyQt5 to load a UI file and set up a window with a radio button. It includes comments for signals and slots, and a main loop at the end. The status bar shows the date and time as 12:16 a.m. 01/03/2025.

```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 import Recursos_rc
4 qtCreatorFile = "P13_RadioButton_Ejemplo1.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): # Emanuel-Izaguirre-Cortes-03
7     def __init__(self): # Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12
13        # Área de los Slots
14    if __name__ == "__main__":
15        app = QtWidgets.QApplication(sys.argv)
16        window = MyApp()
17        window.show()
18        sys.exit(app.exec_())
```

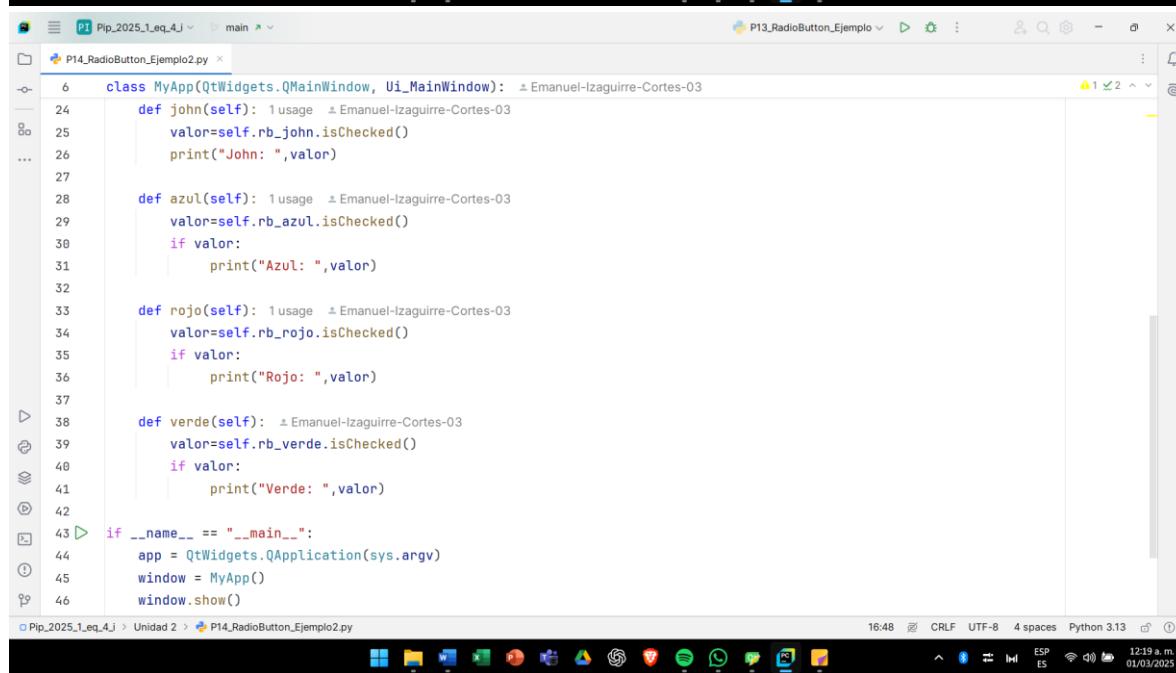




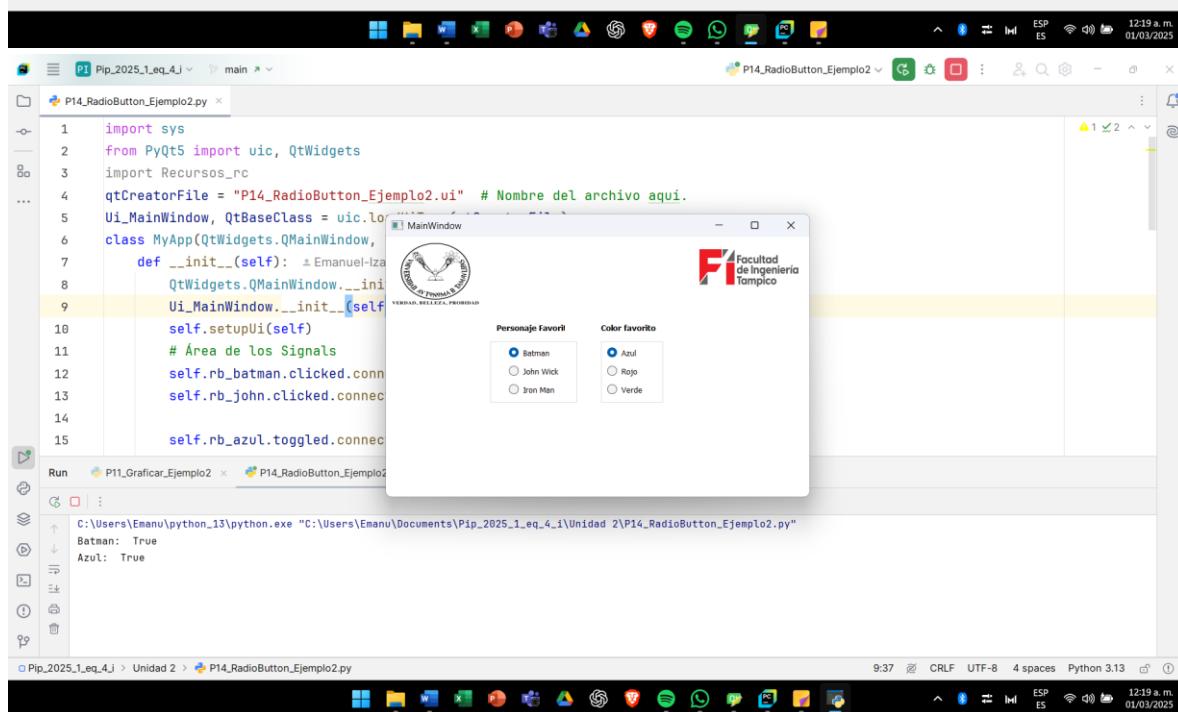
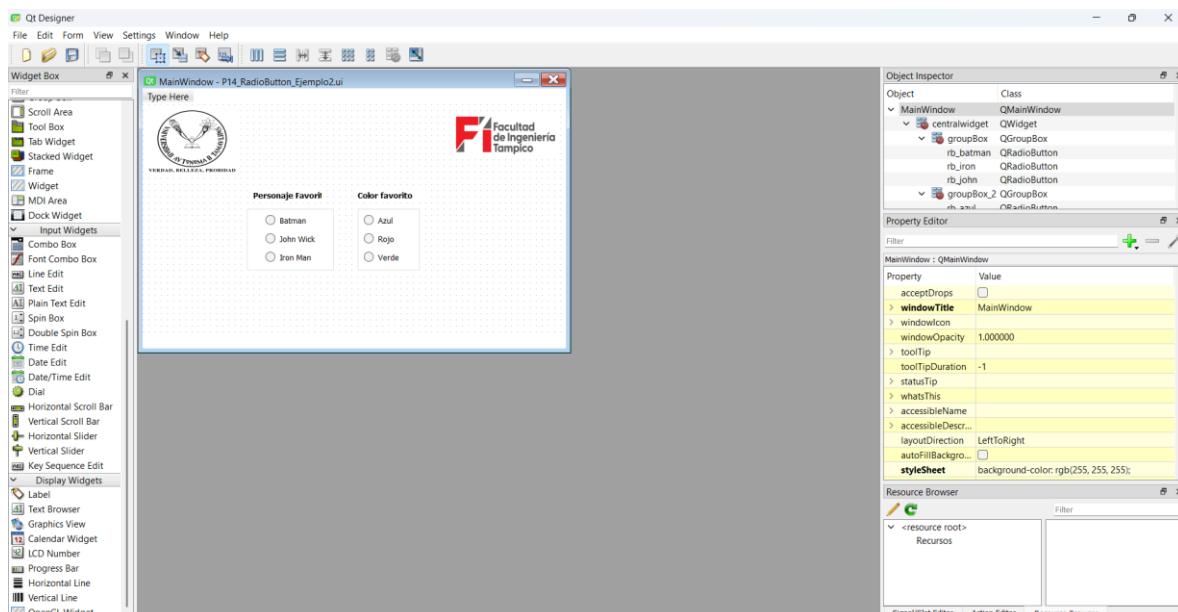
P14_RadioButton_Ejemplo2



```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 import Recursos_rc
4 qtCreatorFile = "P14_RadioButton_Ejemplo2.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     def __init__(self):
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.rb_batman.clicked.connect(self.batman)
13        self.rb_john.clicked.connect(self.john)
14
15        self.rb_azul.toggled.connect(self.azul)
16        self.rb_rojo.toggled.connect(self.rojo)
17
18    # Área de los Slots
19    def batman(self):
20        valor=self.rb_batman.isChecked()
21        print("Batman: ",valor)
22
23    def john(self):
24        valor=self.rb_john.isChecked()
25        print("John: ",valor)
26
27    def azul(self):
28        valor=self.rb_azul.isChecked()
29        if valor:
30            print("Azul: ",valor)
31
32    def rojo(self):
33        valor=self.rb_rojo.isChecked()
34        if valor:
35            print("Rojo: ",valor)
36
37    def verde(self):
38        valor=self.rb_verde.isChecked()
39        if valor:
40            print("Verde: ",valor)
41
42
43 if __name__ == "__main__":
44     app = QtWidgets.QApplication(sys.argv)
45     window = MyApp()
46     window.show()
```



```
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     def john(self):
8         valor=self.rb_john.isChecked()
9         print("John: ",valor)
10
11    def azul(self):
12        valor=self.rb_azul.isChecked()
13        if valor:
14            print("Azul: ",valor)
15
16    def rojo(self):
17        valor=self.rb_rojo.isChecked()
18        if valor:
19            print("Rojo: ",valor)
20
21    def verde(self):
22        valor=self.rb_verde.isChecked()
23        if valor:
24            print("Verde: ",valor)
25
26
27 if __name__ == "__main__":
28     app = QtWidgets.QApplication(sys.argv)
29     window = MyApp()
30     window.show()
```

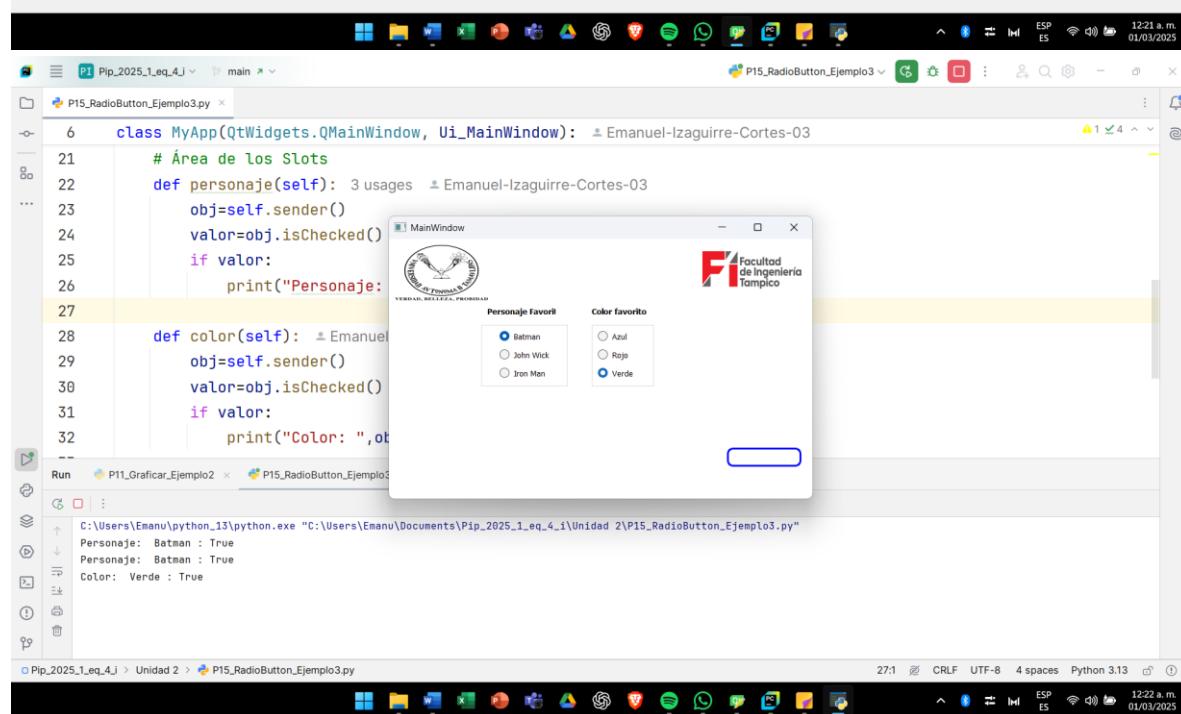
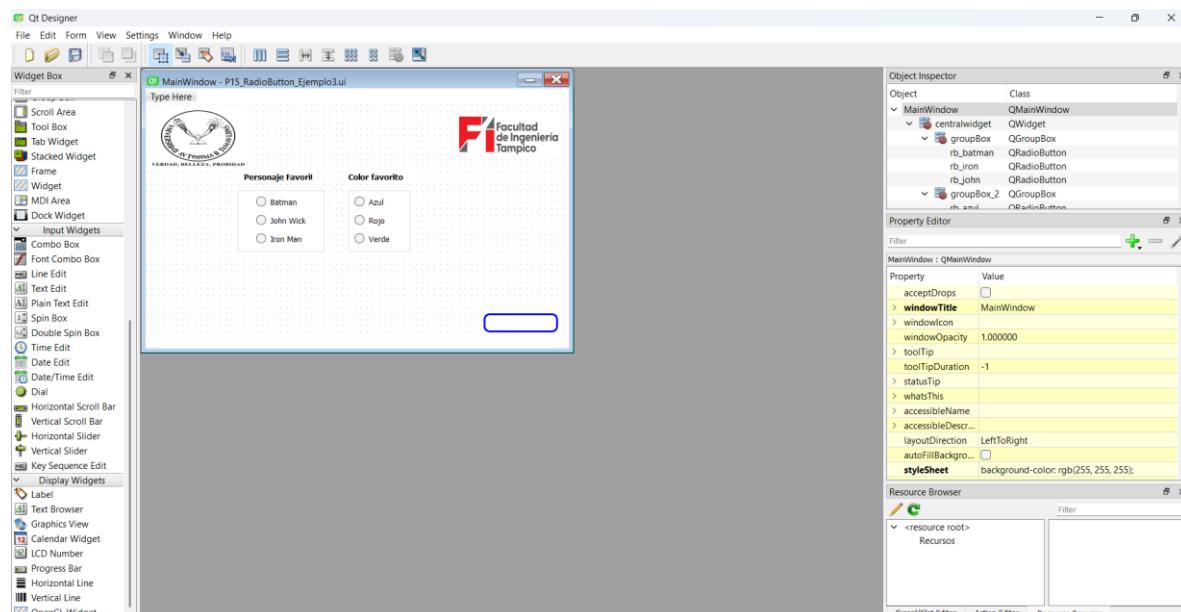




P15_RadioButton_Ejemplo3

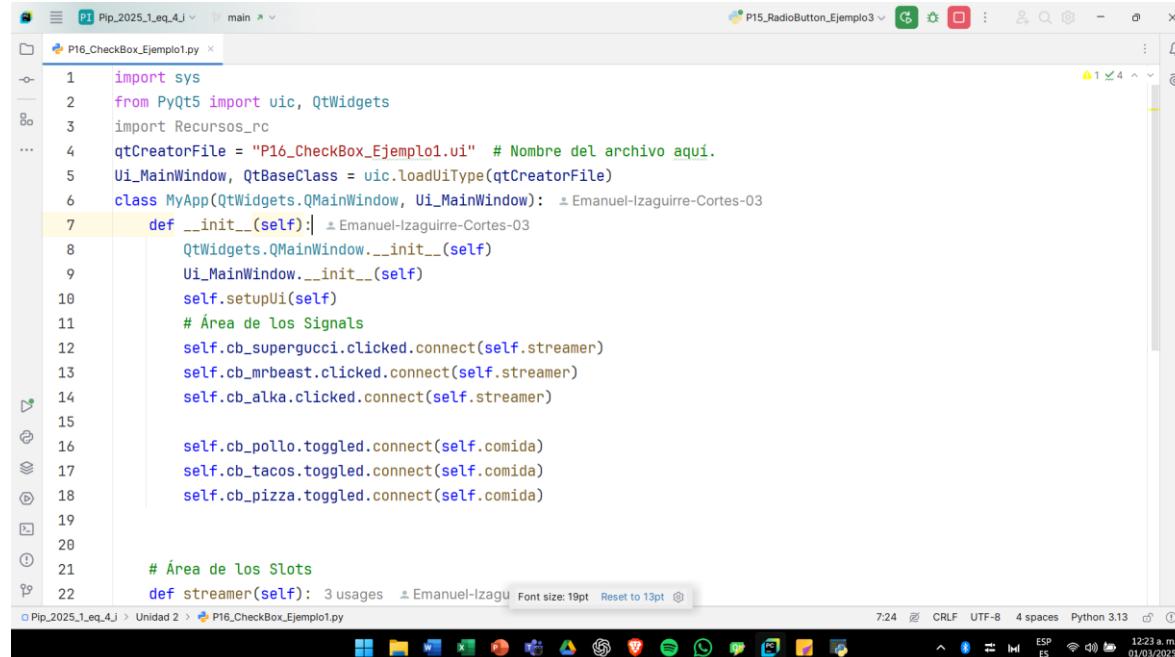
The image shows two side-by-side code editors. Both are displaying the same Python script, `P15_RadioButton_Ejemplo3.py`, which uses the PyQt5 library to create a window with three radio buttons and their corresponding slots.

```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 import Recursos_rc
4 qtCreatorFile = "P15_RadioButton_Ejemplo3.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     def __init__(self): # Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         self.setupUi(self)
10        # Área de los Signals
11        self.rb_batman.clicked.connect(self.personaje)
12        self.rb_john.clicked.connect(self.personaje)
13        self.rb_iron.clicked.connect(self.personaje)
14
15        self.rb_azul.toggled.connect(self.color)
16        self.rb_rojo.toggled.connect(self.color)
17        self.rb_verde.toggled.connect(self.color)
18
19
20    # Área de los Slots
21
22    def personaje(self): 3 usages # Emanuel-Izaguirre-Cortes-03
23        obj=self.sender()
24        valor=obj.isChecked()
25        if valor:
26            print("Personaje: ",obj.text(), ":", valor)
27
28    def color(self): # Emanuel-Izaguirre-Cortes-03
29        obj=self.sender()
30        valor=obj.isChecked()
31        if valor:
32            print("Color: ",obj.text(), ":", valor)
33
34 if __name__ == "__main__":
35     app = QtWidgets.QApplication(sys.argv)
36     window = MyApp()
37     window.show()
38     sys.exit(app.exec_())
```

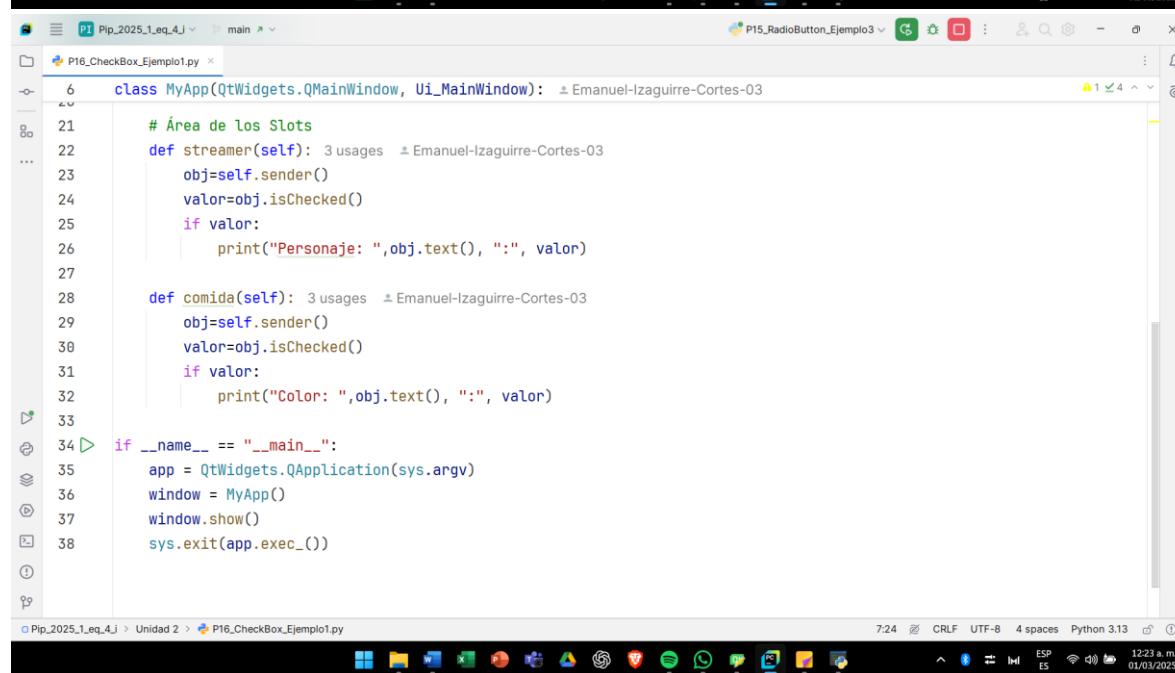




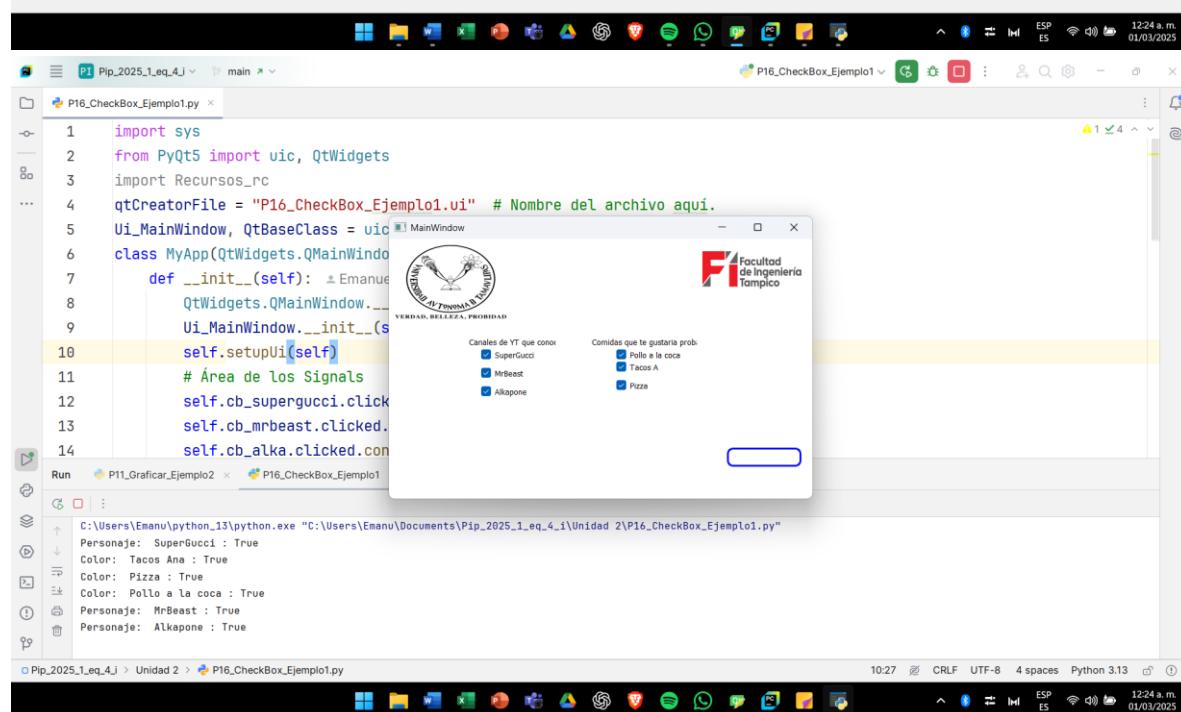
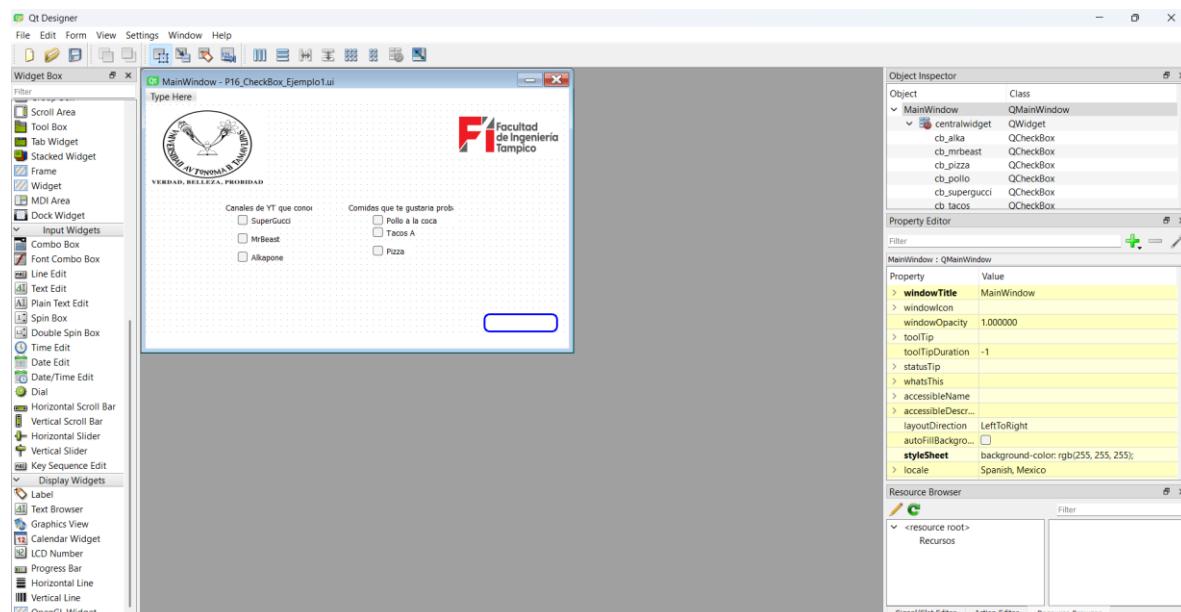
P16_CheckBox_Ejemplo1



```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 import Recursos_rc
4 qtCreatorFile = "P16_CheckBox_Ejemplo1.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     def __init__(self): # Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.cb_supergucci.clicked.connect(self.streamer)
13        self.cb_mrbeast.clicked.connect(self.streamer)
14        self.cb_alka.clicked.connect(self.streamer)
15
16        self.cb_pollo.toggled.connect(self.comida)
17        self.cb_tacos.toggled.connect(self.comida)
18        self.cb_pizza.toggled.connect(self.comida)
19
20
21    # Área de los Slots
22    def streamer(self): 3 usages # Emanuel-Izag
23        obj=self.sender()
24        valor=obj.isChecked()
25        if valor:
26            print("Personaje: ",obj.text(), ":", valor)
27
28    def comida(self): 3 usages # Emanuel-Izaguirre-Cortes-03
29        obj=self.sender()
30        valor=obj.isChecked()
31        if valor:
32            print("Color: ",obj.text(), ":", valor)
33
34    if __name__ == "__main__":
35        app = QtWidgets.QApplication(sys.argv)
36        window = MyApp()
37        window.show()
38        sys.exit(app.exec_())
39
```

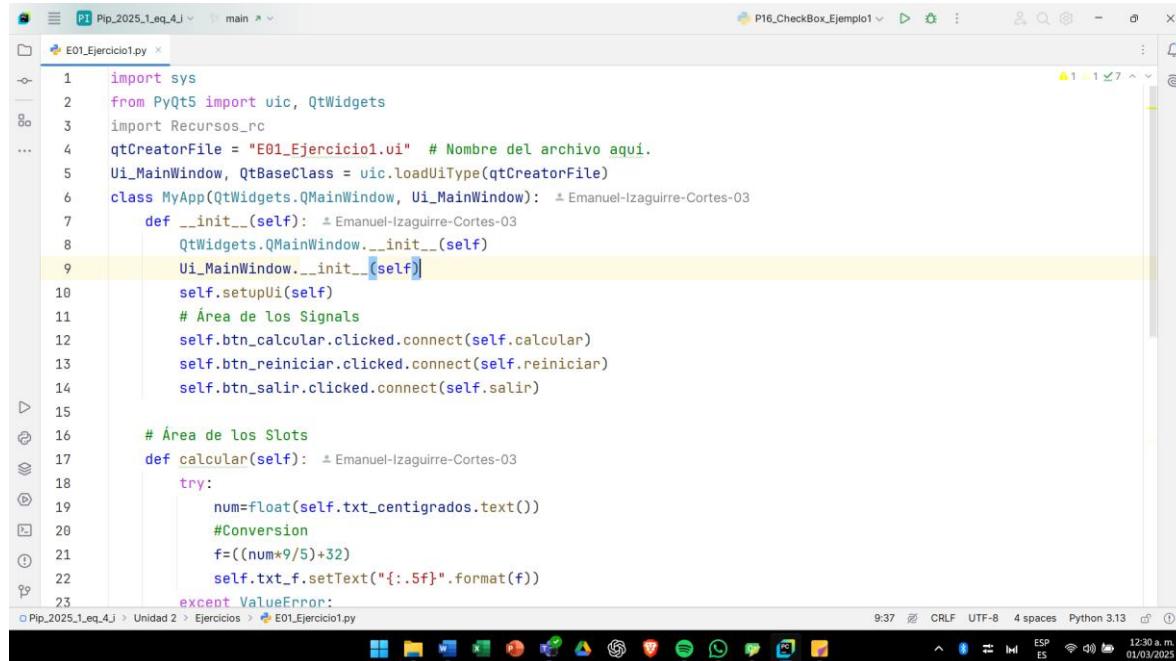


```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 import Recursos_rc
4 qtCreatorFile = "P16_CheckBox_Ejemplo1.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     def __init__(self): # Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.cb_supergucci.clicked.connect(self.streamer)
13        self.cb_mrbeast.clicked.connect(self.streamer)
14        self.cb_alka.clicked.connect(self.streamer)
15
16        self.cb_pollo.toggled.connect(self.comida)
17        self.cb_tacos.toggled.connect(self.comida)
18        self.cb_pizza.toggled.connect(self.comida)
19
20
21    # Área de los Slots
22    def streamer(self): 3 usages # Emanuel-Izag
23        obj=self.sender()
24        valor=obj.isChecked()
25        if valor:
26            print("Personaje: ",obj.text(), ":", valor)
27
28    def comida(self): 3 usages # Emanuel-Izaguirre-Cortes-03
29        obj=self.sender()
30        valor=obj.isChecked()
31        if valor:
32            print("Color: ",obj.text(), ":", valor)
33
34    if __name__ == "__main__":
35        app = QtWidgets.QApplication(sys.argv)
36        window = MyApp()
37        window.show()
38        sys.exit(app.exec_())
39
```



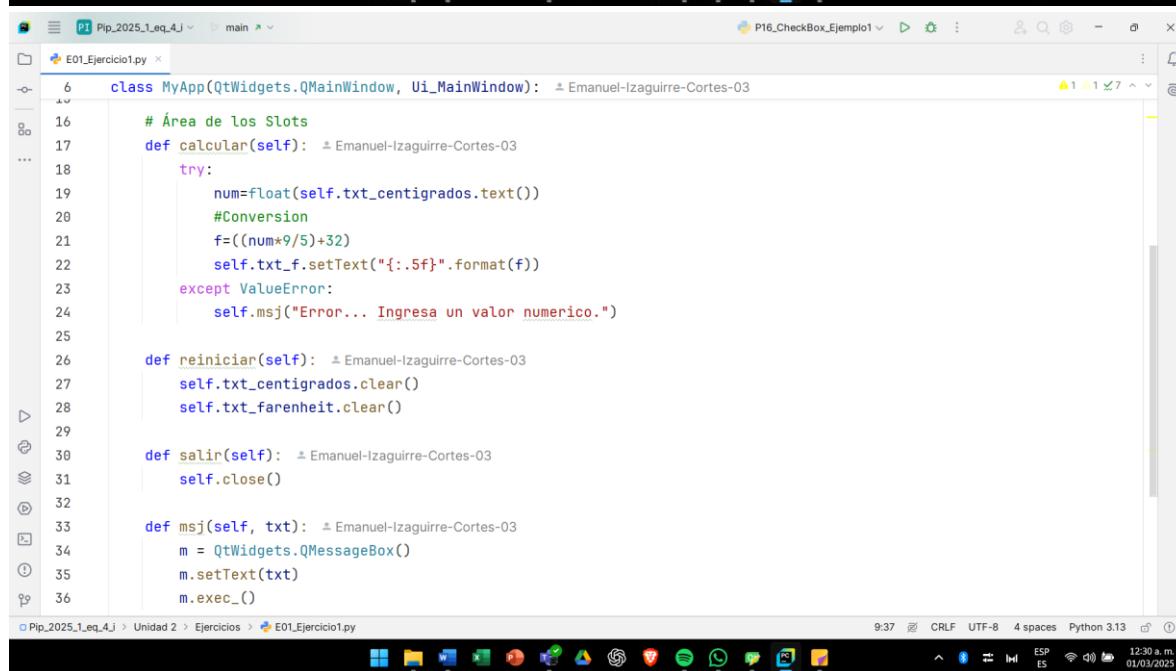


E01_ Cambio de Grados Centígrados Fahrenheit



```
1 import sys
2 from PyQt5 import uic, QtWidgets
3 import Recursos_rc
4 qtCreatorFile = "E01_Ejercicio1.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     def __init__(self): # Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.btn_calcular.clicked.connect(self.calcular)
13        self.btn_reiniciar.clicked.connect(self.reiniciar)
14        self.btn_salir.clicked.connect(self.salir)
15
16        # Área de los Slots
17    def calcular(self): # Emanuel-Izaguirre-Cortes-03
18        try:
19            num=float(self.txt_centigrados.text())
20            #Conversion
21            f=((num*9/5)+32)
22            self.txt_f.setText("{:.5f}".format(f))
23        except ValueError:
24            self.msg("Error... Ingresá un valor numérico.")
25
26    def reiniciar(self): # Emanuel-Izaguirre-Cortes-03
27        self.txt_centigrados.clear()
28        self.txt_fahrenheit.clear()
29
30    def salir(self):
31        self.close()
32
33    def msg(self, txt): # Emanuel-Izaguirre-Cortes-03
34        m = QtWidgets.QMessageBox()
35        m.setText(txt)
36        m.exec_()

```



```
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
7     # Área de los Slots
8     def calcular(self): # Emanuel-Izaguirre-Cortes-03
9         try:
10            num=float(self.txt_centigrados.text())
11            #Conversion
12            f=((num*9/5)+32)
13            self.txt_f.setText("{:.5f}".format(f))
14        except ValueError:
15            self.msg("Error... Ingresá un valor numérico.")
16
17    def reiniciar(self): # Emanuel-Izaguirre-Cortes-03
18        self.txt_centigrados.clear()
19        self.txt_fahrenheit.clear()
20
21    def salir(self):
22        self.close()
23
24    def msg(self, txt): # Emanuel-Izaguirre-Cortes-03
25        m = QtWidgets.QMessageBox()
26        m.setText(txt)
27        m.exec_()

```



The screenshot shows the Qt Designer interface for a Python application. The code editor window displays a Python script named E01_Ejercicio1.py, which contains the implementation of a conversion application. The Qt Designer window shows the graphical user interface (GUI) with a logo, a title bar, and several buttons and labels. The Object Inspector, Property Editor, and Resource Browser panels are also visible.

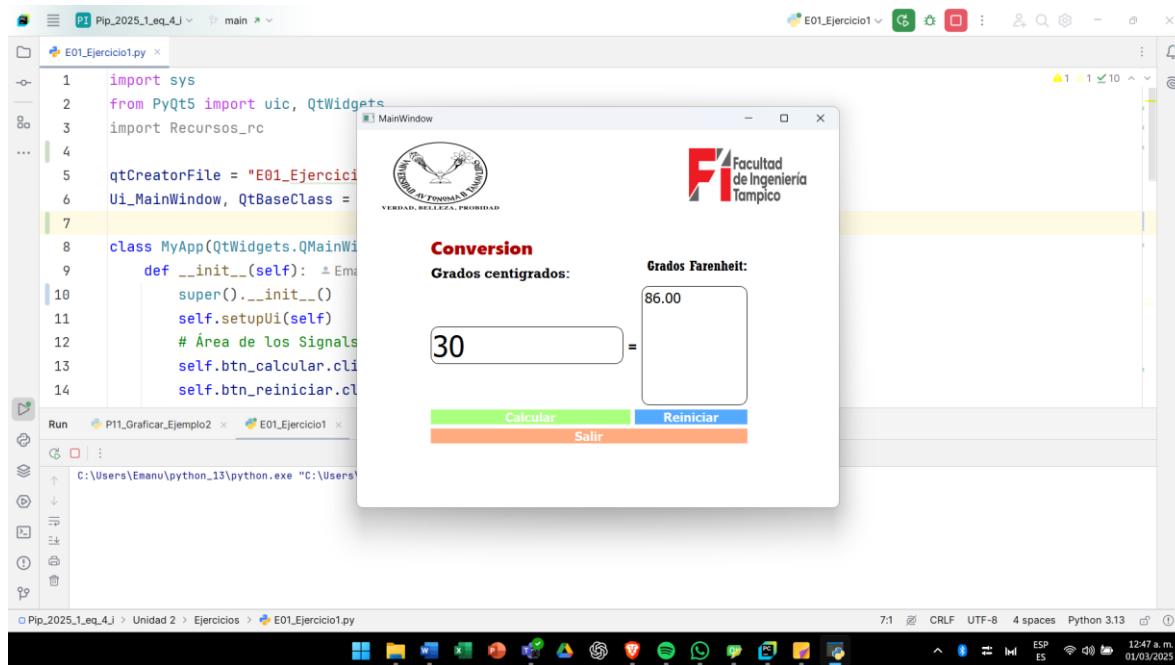
```
class MyApp(QtWidgets.QMainWindow, Ui_MainWindow):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.setWindowTitle("Conversion")
        self.setCentralWidget(self.gridLayout)
        self.reiniciar.clicked.connect(self.reiniciar)
        self.sair.clicked.connect(self.sair)
        self.calcular.clicked.connect(self.calcular)
        self.reinicar.clicked.connect(self.reinicar)

    def reiniciar(self):
        self.txt_centigrados.clear()
        self.txt_farenheit.clear()

    def sair(self):
        self.close()

    def msj(self, txt):
        m = QtWidgets.QMessageBox()
        m.setText(txt)
        m.exec_()

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    window = MyApp()
    window.show()
    sys.exit(app.exec_())
```



Este ejercicio es una **aplicación de escritorio en PyQt5** que permite convertir una temperatura de **grados Celsius a Fahrenheit**.

◆ **Descripción del Programa:**

El usuario ingresa una temperatura en **grados Celsius** en un campo de texto y, al presionar el botón "**Calcular**", la aplicación realiza la conversión a **grados Fahrenheit** y muestra el resultado en otro campo de texto.

También incluye botones para **reiniciar** los campos y otro para **cerrar** la aplicación.

◆ **Funcionalidad del Código:**

1. **Interfaz Gráfica (GUI):**

- Se diseña con **Qt Designer** (E01_Ejercicio1.ui).
- Contiene botones, etiquetas y cuadros de texto para la conversión.

2. **Conversión Matemática:**

- Fórmula: $F = (C \times 9/5) + 32$
- Donde C es la temperatura en Celsius ingresada por el usuario.

3. **Interacción del Usuario:**

- **Botón "Calcular":** Convierte la temperatura y la muestra.
- **Botón "Reiniciar":** Limpia los cuadros de texto.
- **Botón "Salir":** Cierra la aplicación.

4. **Manejo de Errores:**

- Si el usuario ingresa un valor no numérico, aparece un **mensaje de error**.

- ◆ **Ejemplo de Uso:**
 - ◆ Si el usuario ingresa 25 en **Celsius** y presiona "**Calcular**", el programa mostrará 77.00 en **Fahrenheit**.
 - ◆ Si el usuario ingresa abc, aparecerá un mensaje de error: "**Error... Ingresa un valor numérico.**"

Este ejercicio es útil para practicar el uso de PyQt5, la manipulación de interfaces gráficas y la conversión de datos numéricos.

E02_Conversor de Horas (En formato de 24horas) a Segundos del día



```
1 import sys
2 from PyQt5 import uic, QtWidgets
3
4 qtCreatorFile = "E02_ConversorHoras.ui" # Nombre del archivo .ui
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6
7
8 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): new *
9     def __init__(self): new *
10         super().__init__()
11         self.setupUi(self)
12         # Conexión de botones a funciones
13         self.btn_calcular.clicked.connect(self.calcular)
14         self.btn_reiniciar.clicked.connect(self.reiniciar)
15         self.btn_salir.clicked.connect(self.salir)
16
17     def calcular(self): new *
18         try:
19             # Obtener la hora ingresada
20             tiempo = self.txt_hora.text() # Formato esperado HH:MM:SS
21             partes = tiempo.split(":")
22
23             if len(partes) != 3:
24                 raise ValueError # Si el formato es incorrecto
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
```

Font size: 15pt Reset to 13pt



```
class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): new *
    def calcular(self): new *
        if len(partes) != 3:
            raise ValueError # Si el formato es incorrecto

        horas = int(partes[0])
        minutos = int(partes[1])
        segundos = int(partes[2])

        # Validaciones básicas
        if not (0 <= horas < 24 and 0 <= minutos < 60 and 0 <= segundos < 60):
            raise ValueError

        # Convertir a segundos
        total_segundos = (horas * 3600) + (minutos * 60) + segundos

        # Mostrar resultado
        self.txt_segundos.setText(str(total_segundos))

    except ValueError:
        self.msj("Error... Ingresa una hora válida en formato HH:MM:SS")

    def reiniciar(self): new *
        self.txt_hora.clear()
        self.txt_segundos.clear()

Run E02_Ejercicio2
```

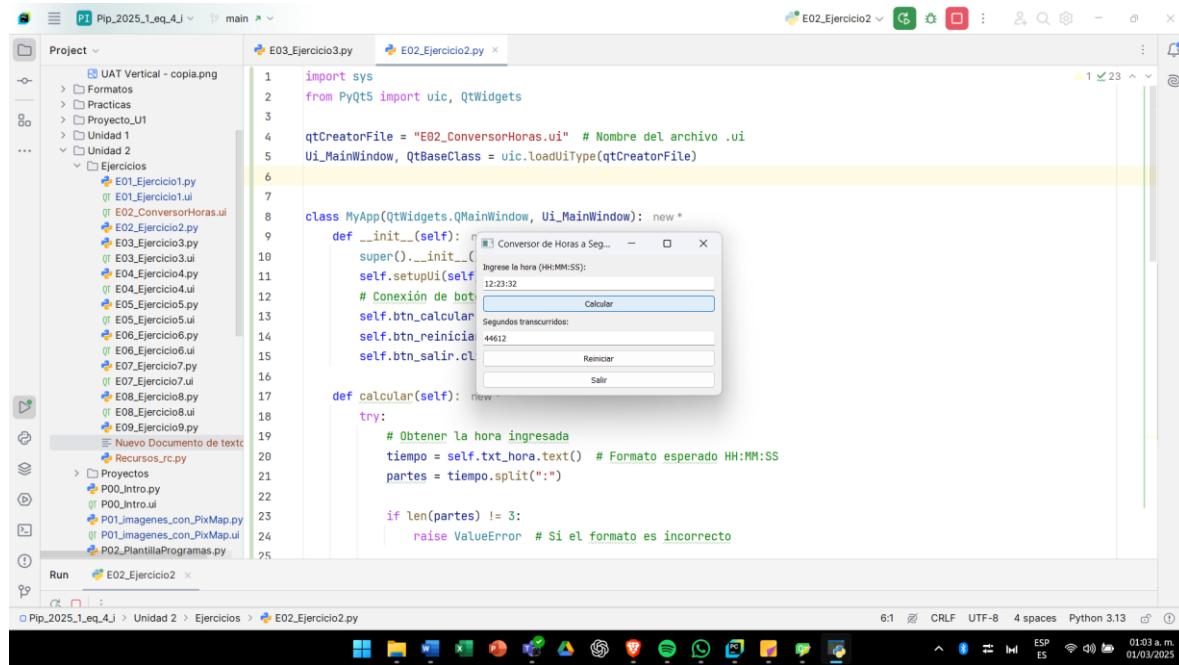
```
self.msj("Error... Ingresa una hora válida en formato HH:MM:SS")

def reiniciar(self): new *
    self.txt_hora.clear()
    self.txt_segundos.clear()

def salir(self): new *
    self.close()

def msj(self, txt): new *
    m = QtWidgets.QMessageBox()
    m.setText(txt)
    m.exec_()

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    window = MyApp()
    window.show()
    sys.exit(app.exec_())
```



The screenshot shows the PyCharm IDE interface. The project tree on the left lists several files under 'Project' and 'Ejercicios'. The code editor window displays Python code for a timer application. The code imports sys and PyQt5, loads a UI file named E02_ConversorHoras.ui, and defines a class MyApp that inherits from QMainWindow and Ui_MainWindow. It includes methods for initializing the window, calculating time differences, and handling button events like 'Calcular', 'Reiniciar', and 'Salir'. A preview of the application's GUI is shown on the right, featuring a text input for time, a 'Calcular' button, and two output fields for seconds and milliseconds.

Resumen del Ejercicio: Conversor de Horas a Segundos

✓ Objetivo: Convertir una hora en formato HH:MM:SS a segundos transcurridos en el día usando una interfaz gráfica con PyQt5.

✓ Funcionamiento:

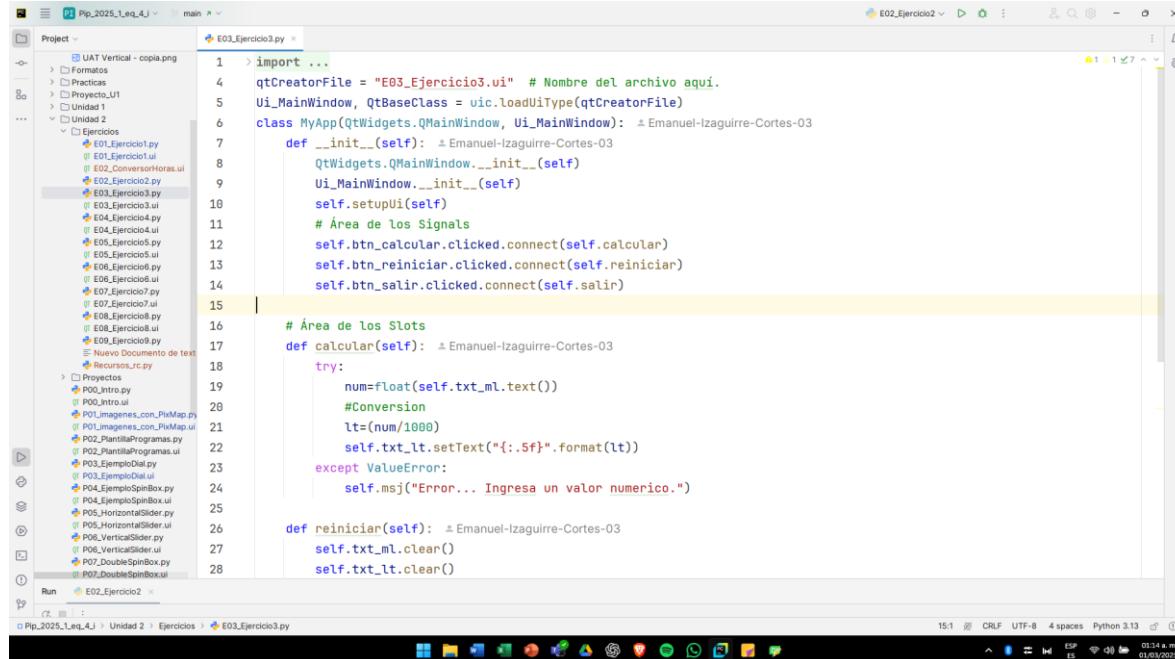
1. **El usuario ingresa una hora** en txt_hora.
2. **Presiona "Calcular"** y el programa convierte la hora a segundos:
segundos=(horas×3600)+(minutos×60)+segundos\text{segundos} = (\text{horas} \times 3600) + (\text{minutos} \times 60) + \text{segundos}
3. **El resultado aparece en txt_segundos.**
4. **Manejo de errores:** Si el formato es incorrecto, se muestra un mensaje de error.
5. **Botones adicionales:**
 - **"Reiniciar"**: Borra los campos.
 - **"Salir"**: Cierra la aplicación.

✓ Ejemplo:

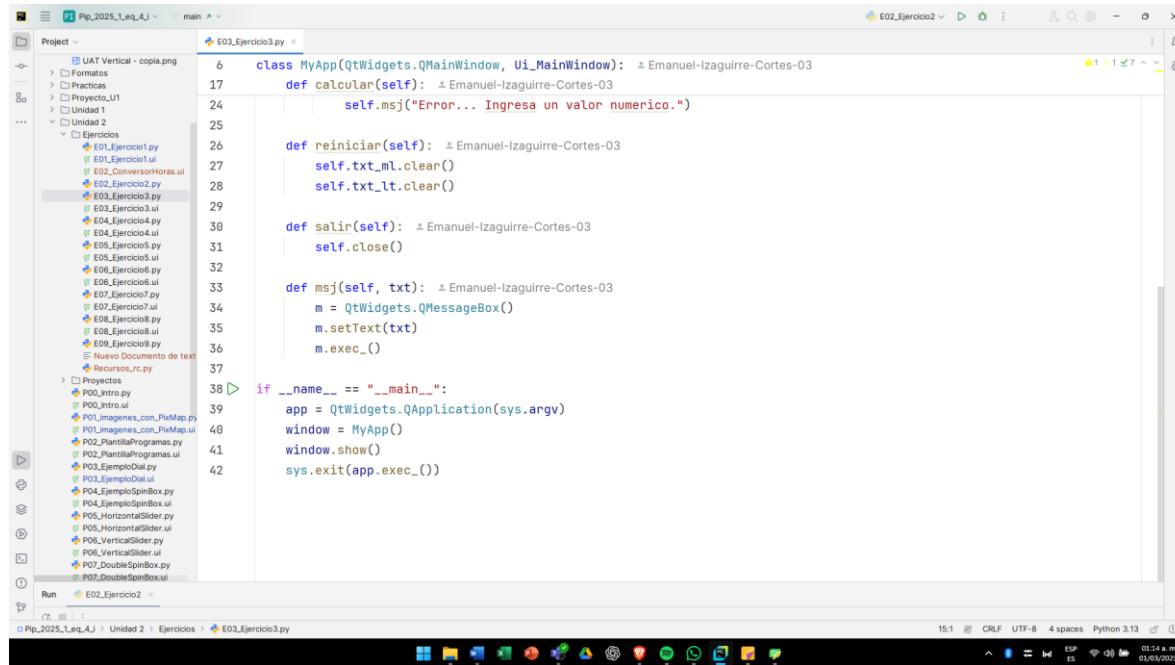
- Entrada: 12:30:15 → Salida: 45015 segundos.
- Entrada inválida: 25:00:00 → Muestra error.



E03_ Mililitros a litros



```
1 > import ...
4 qtCreatorFile = "E03_Ejercicio3.ui" # Nombre del archivo aqui.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): □ Emanuel-Izaguirre-Cortes-03
7     def __init__(self): □ Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.btn_calcular.clicked.connect(self.calcular)
13        self.btn_reiniciar.clicked.connect(self.reiniciar)
14        self.btn_salir.clicked.connect(self.salir)
15
16        # Área de los Slots
17        def calcular(self): □ Emanuel-Izaguirre-Cortes-03
18            try:
19                num=float(self.txt_ml.text())
20                #Conversion
21                lt=(num/1000)
22                self.txt_lt.setText("{:.5f}".format(lt))
23            except ValueError:
24                self.msg("Error... Ingresa un valor numerico.")
25
26        def reiniciar(self): □ Emanuel-Izaguirre-Cortes-03
27            self.txt_ml.clear()
28            self.txt_lt.clear()
29
30        def salir(self): □ Emanuel-Izaguirre-Cortes-03
31            self.close()
32
33        def msg(self, txt): □ Emanuel-Izaguirre-Cortes-03
34            m = QtWidgets.QMessageBox()
35            m.setText(txt)
36            m.exec_()
37
38    if __name__ == "__main__":
39        app = QtWidgets.QApplication(sys.argv)
40        window = MyApp()
41        window.show()
42        sys.exit(app.exec_())
```



```
6    class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): □ Emanuel-Izaguirre-Cortes-03
17        def calcular(self): □ Emanuel-Izaguirre-Cortes-03
18            self.msg("Error... Ingresa un valor numerico.")
19
20        def reiniciar(self): □ Emanuel-Izaguirre-Cortes-03
21            self.txt_ml.clear()
22            self.txt_lt.clear()
23
24        def salir(self): □ Emanuel-Izaguirre-Cortes-03
25            self.close()
26
27        def msg(self, txt): □ Emanuel-Izaguirre-Cortes-03
28            m = QtWidgets.QMessageBox()
29            m.setText(txt)
30            m.exec_()
31
32
33    if __name__ == "__main__":
34        app = QtWidgets.QApplication(sys.argv)
35        window = MyApp()
36        window.show()
37        sys.exit(app.exec_())
```



The screenshot shows the PyCharm IDE interface. On the left is the project tree with files like UAT Vertical - copia.png, Ejercicios, and various .py files for exercises. The main editor window displays the Python code for E03_Ejercicio3.py. The code defines a class MyApp with methods calcular, reiniciar, and salir. It includes UI logic for a conversion application. A preview window shows the application's UI with a logo, a title 'Conversion', two input fields ('Millilitros:' and 'Litros:'), and three buttons ('Calcular', 'Reiniciar', and 'Salir').

The screenshot shows the Qt Designer interface. It displays the 'MainWindow - E03_Ejercicio3.ui' window. The UI consists of a logo at the top, followed by a title 'Conversion'. Below it are two input fields: 'Millilitros:' and 'Litros:', separated by an equals sign (=). At the bottom are three buttons: 'Calcular', 'Reiniciar', and 'Salir'. To the right of the designer are the Object Inspector, Property Editor, and Resource Browser panels.

Tu código es un **convertidor de mililitros a litros** usando **PyQt5**.

📌 **Explicación sencilla:**

1. **Carga la interfaz gráfica** desde el archivo E03_Ejercicio3.ui.
2. **Conecta los botones** con sus respectivas funciones:
 - o "Calcular" → Convierte mililitros a litros.
 - o "Reiniciar" → Borra los campos de entrada.
 - o "Salir" → Cierra la aplicación.

3. Función calcular():

- Toma el número ingresado en txt_ml.
 - Lo convierte a litros (mililitros \div 1000).
 - Muestra el resultado en txt_lt.
 - Si el usuario ingresa texto en lugar de un número, muestra un **mensaje de error**.

4. **Función msj()**: Muestra un cuadro de diálogo con un mensaje de error si la entrada es incorrecta.

Ejemplo de uso:

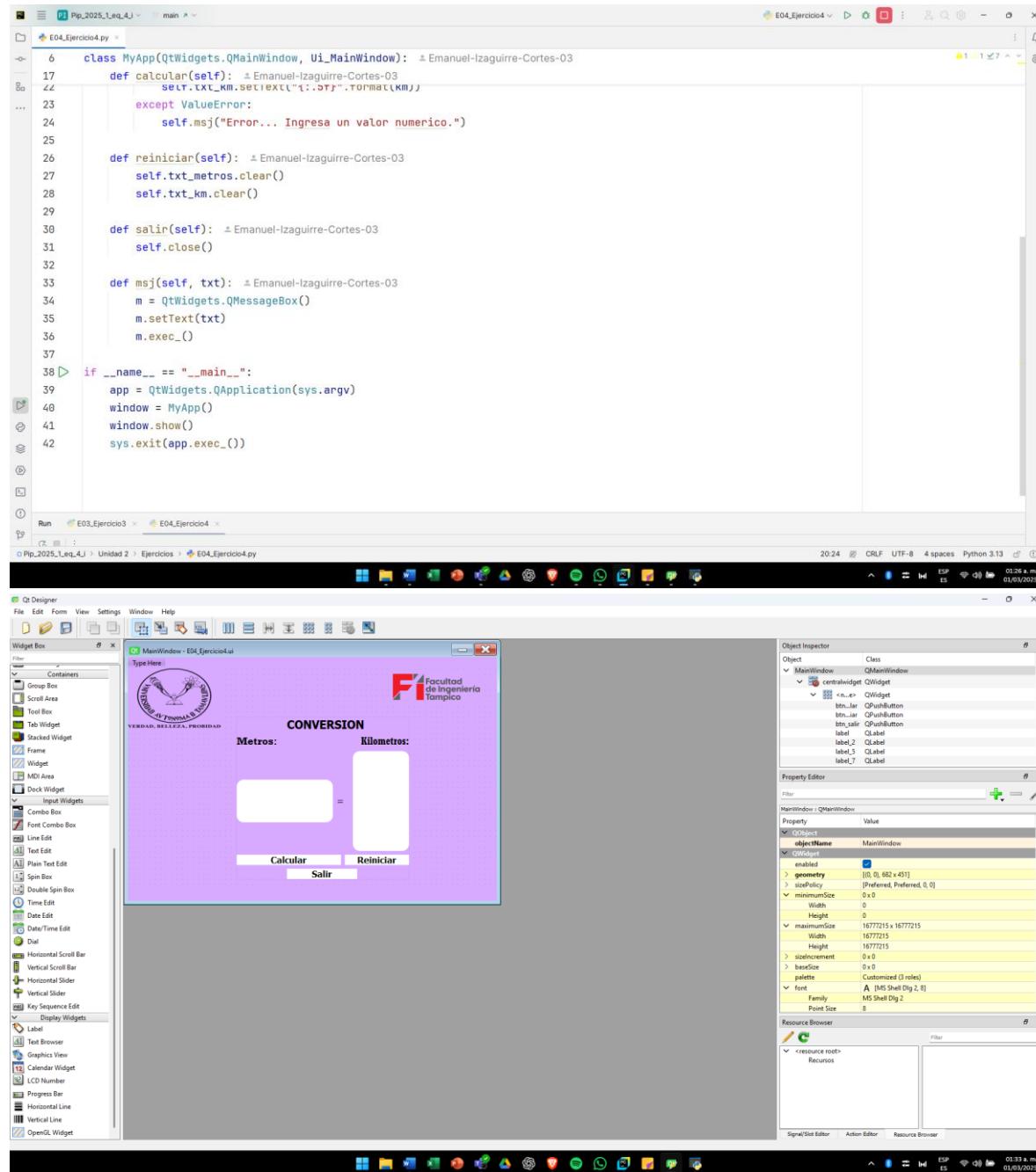
- Entrada: 2500 ml → Salida: 2.50000 litros
 - Entrada inválida: "abc" → **Muestra error**

Es un programa simple y funcional para convertir unidades de volumen.

E04_ Metros a Kilómetros

The screenshot shows the PyCharm IDE interface with the following details:

- Title Bar:** Pip_2025_1_leq_4_J, main, E04_Ejercicio4.
- Code Editor:** The file E04_Ejercicio4.py contains Python code for a PyQt5 application. The code defines a class MyApp that inherits from QMainWindow and Ui_MainWindow. It includes methods for calculating distance from meters to kilometers, clearing text inputs, and handling errors. A specific line of code `km=(num/1000)` is highlighted in yellow.
- Toolbars and Status Bar:** Standard PyCharm toolbars and status bar showing the current file, line number (20:24), encoding (CRLF), and Python version (Python 3.13).
- Bottom Navigation:** Shows the project structure: Pip_2025_1_leq_4_J > Unidad 2 > Ejercicios > E04_Ejercicio4.py.



The screenshot shows the Qt Designer interface with the following details:

- Code Editor:** Displays the Python code for the application, specifically the file E04_Ejercicio4.py. The code defines a class MyApp that inherits from QMainWindow. It includes methods for calculating conversions, restarting the application, and exiting. It also handles errors and message boxes.
- Qt Designer Window:** Shows the visual design of the application window titled "E04_Ejercicio4". The window has a purple background and contains:
 - A logo for "Facultad de Ingeniería Tampico" at the top right.
 - A circular emblem with the text "VERDAD, BELLEZA, PROBIDAD" at the bottom left.
 - A title bar with the text "CONVERSION".
 - Two input fields labeled "Metros" and "Kilometros" with an equals sign between them.
 - Three buttons at the bottom: "Calcular", "Reiniciar", and "Salir".
- Object Inspector:** A sidebar showing the object hierarchy. It lists the MainWindow, centralwidget, and various child widgets like buttons and labels.
- Property Editor:** A sidebar showing properties for the selected object. For example, the geometry is set to [0, 0, 682 x 451].
- Resource Browser:** A sidebar showing resources available in the project.

The screenshot shows a Python IDE interface with two main windows. The left window is a code editor for 'E04_Ejercicio4.py' containing Python code for a conversion application. The right window is a graphical user interface (GUI) titled 'CONVERSION' with a purple background. It features a logo for 'Facultad de Ingeniería Tampico' and two input fields: 'Metros:' (containing '6700') and 'Kilometros:' (containing '6.7000'). Below these is a large digital display showing '6700 =' with buttons for 'Calcular', 'Reiniciar', and 'Salir'. The status bar at the bottom of the IDE shows the file path 'D:\Pip_2025_1_leq_4_j\Unidad 2\Ejercicios\E04_Ejercicio4.py' and the system time '12:57'.

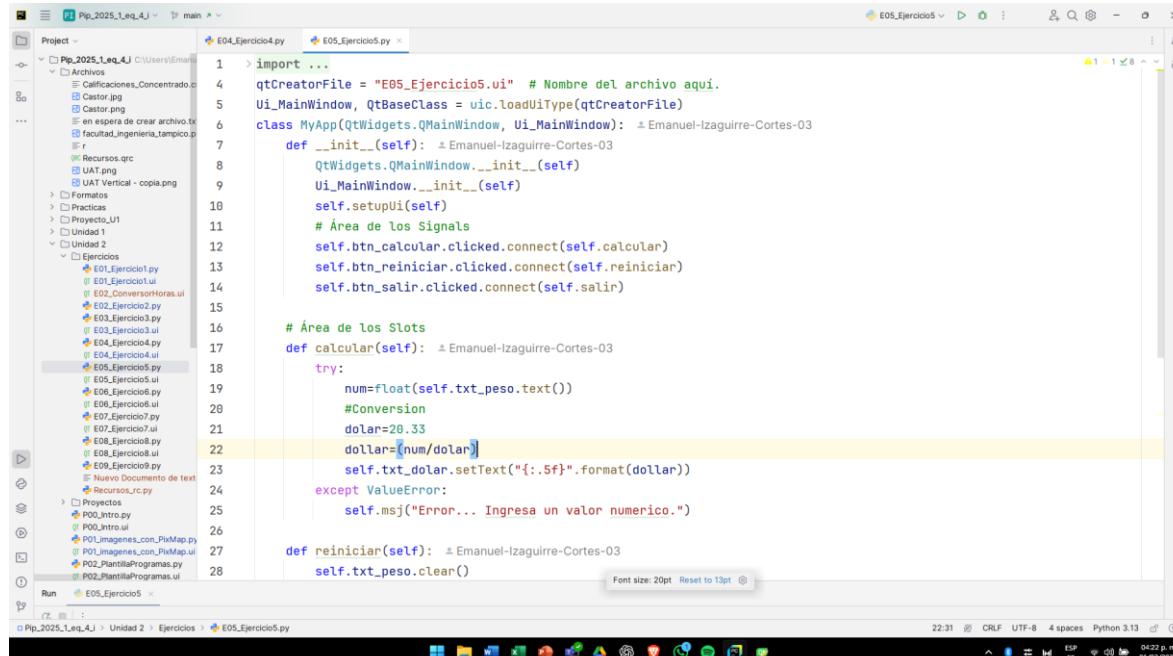
```
1 > import ...
4 qtCreatorFile = "E04_Ejercicio4.ui" # Nombre del archivo aqui.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): ± Emanuel-Izaguirre-Cortes-03
7     def __init__(self): ± Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.btn_calcular.clicked.connect(self.calcular)
13        self.btn_reiniciar.clicked.connect(self.reiniciar)
14        self.btn_salir.clicked.connect(self.salir)
15
16        # Área de los Slots
17    def calcular(self): ± Emanuel-Izaguirre-Cortes-03
18        try:
19            num=float(self.txt_metros.text())
20            #Conversion
21            km=(num/1000)
22            self.txt_km.setText("{:.5f}".format(km))
23        except ValueError:
24            self.msj("Error... Ingresá un valor numerico.")
25
26    def reiniciar(self): ± Emanuel-Izaguirre-Cortes-03
27        self.txt_metros.clear()
28        self.txt_km.clear()
```

El programa es un **convertidor de metros a kilómetros** usando PyQt5.

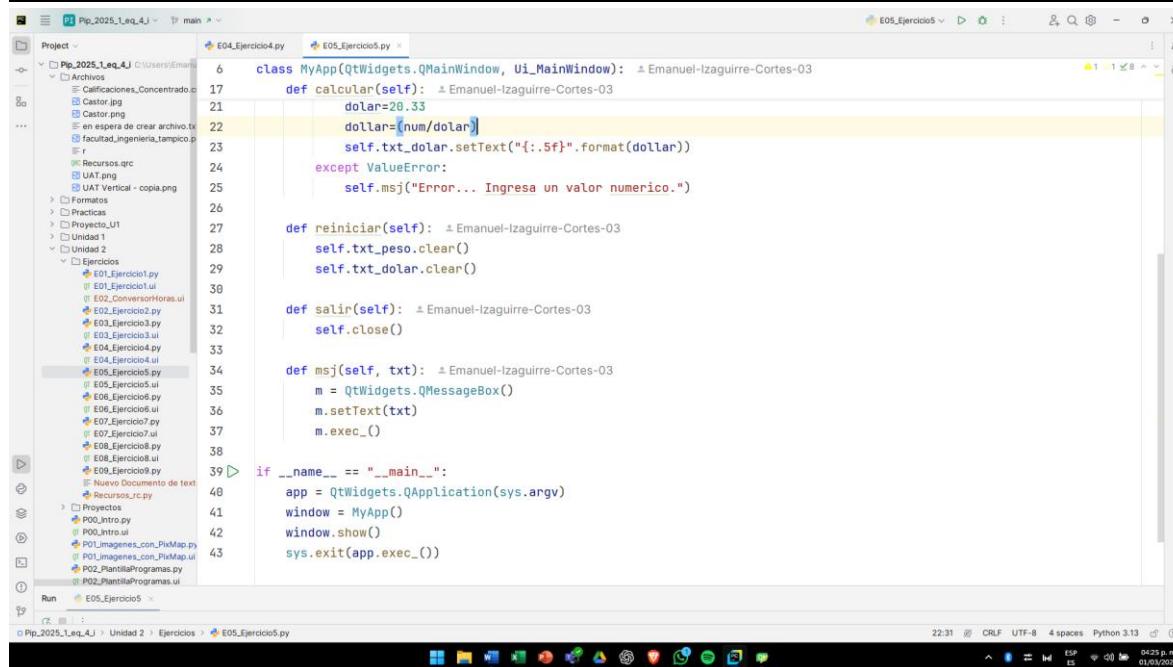
1. El usuario ingresa un valor en **metros**.
 2. Al presionar el botón "**Calcular**", el programa convierte los metros a kilómetros ($\text{metros} \div 1000$) y muestra el resultado.
 3. Si se ingresa un valor no numérico, se muestra un **mensaje de error**.
 4. El botón "**Reiniciar**" borra los campos, y "**Salir**" cierra la aplicación.



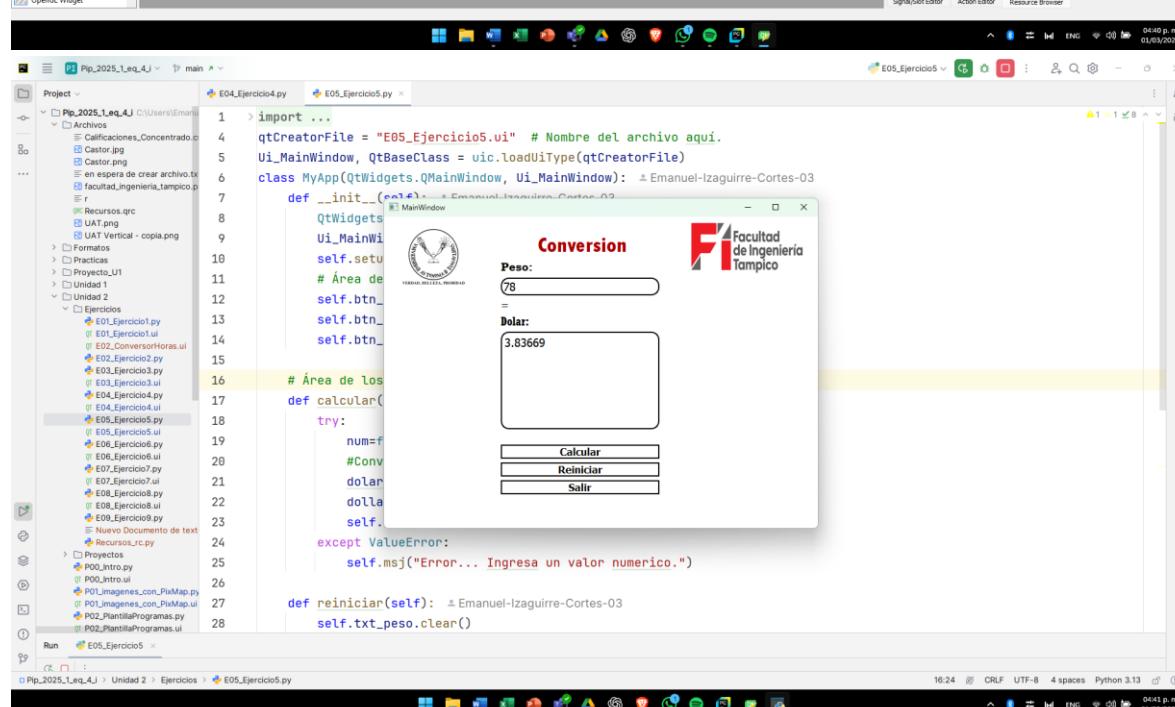
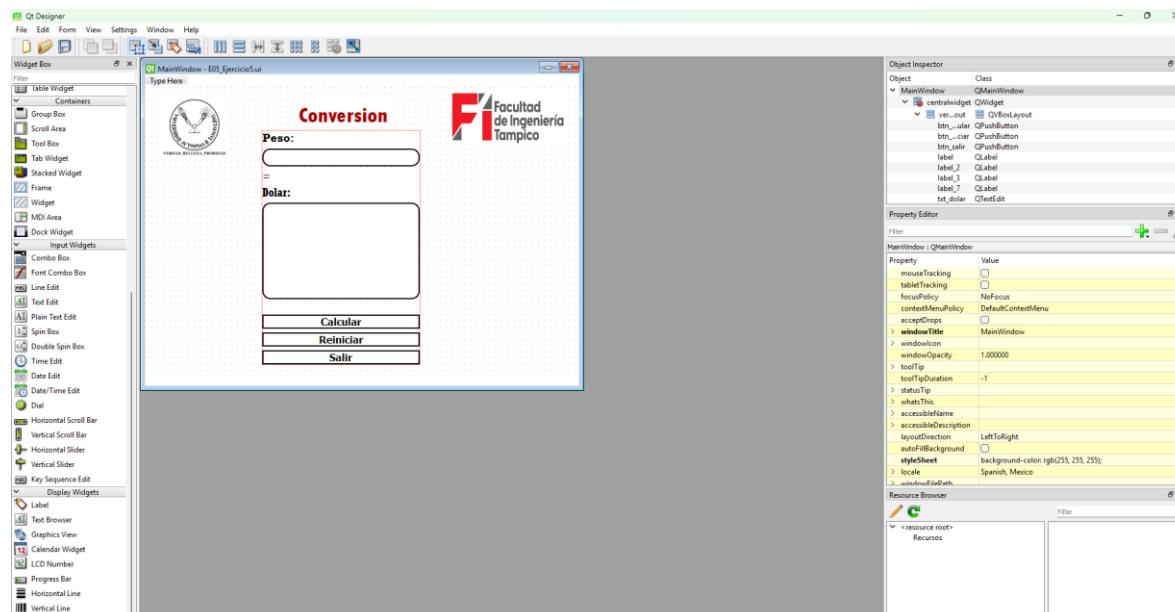
E05_ Peso a dólar estadounidense



```
1 > import ...
2
3 qtCreatorFile = "E05_Ejercicio5.ui" # Nombre del archivo aqui.
4
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6
7 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): □ Emanuel-Izaguirre-Cortes-03
8     def __init__(self): □ Emanuel-Izaguirre-Cortes-03
9         QtWidgets.QMainWindow.__init__(self)
10        self.setupUi(self)
11
12        # Área de los Signals
13        self.btn_calcular.clicked.connect(self.calcular)
14        self.btn_reiniciar.clicked.connect(self.reiniciar)
15        self.btn_salir.clicked.connect(self.salir)
16
17        # Área de los Slots
18    def calcular(self): □ Emanuel-Izaguirre-Cortes-03
19        try:
20            num=float(self.txt_peso.text())
21            #Conversion
22            dolar=20.33
23            dollar=[num/dolar]
24            self.txt_dolar.setText("{:.5f}".format(dollar))
25        except ValueError:
26            self.msj("Error... Ingresar un valor numerico.")
27
28    def reiniciar(self): □ Emanuel-Izaguirre-Cortes-03
29        self.txt_peso.clear()
30
31    def salir(self):
32        self.close()
33
34    def msj(self, txt): □ Emanuel-Izaguirre-Cortes-03
35        m = QtWidgets.QMessageBox()
36        m.setText(txt)
37        m.exec_()
38
39    if __name__ == "__main__":
40        app = QtWidgets.QApplication(sys.argv)
41        window = MyApp()
42        window.show()
43        sys.exit(app.exec_())
```



```
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): □ Emanuel-Izaguirre-Cortes-03
7     def calcular(self): □ Emanuel-Izaguirre-Cortes-03
8         dolar=20.33
9         dollar=[num/dolar]
10        self.txt_dolar.setText("{:.5f}".format(dollar))
11
12    def reiniciar(self): □ Emanuel-Izaguirre-Cortes-03
13        self.txt_peso.clear()
14        self.txt_dolar.clear()
15
16    def salir(self):
17        self.close()
18
19    def msj(self, txt): □ Emanuel-Izaguirre-Cortes-03
20        m = QtWidgets.QMessageBox()
21        m.setText(txt)
22        m.exec_()
23
24    if __name__ == "__main__":
25        app = QtWidgets.QApplication(sys.argv)
26        window = MyApp()
27        window.show()
28        sys.exit(app.exec_())
```



El programa es una aplicación de escritorio desarrollada en Python con PyQt5 que convierte pesos mexicanos a dólares. Se carga una interfaz gráfica desde un archivo .ui, y se define una clase MyApp que hereda de QMainWindow y la interfaz generada. La aplicación tiene tres botones:

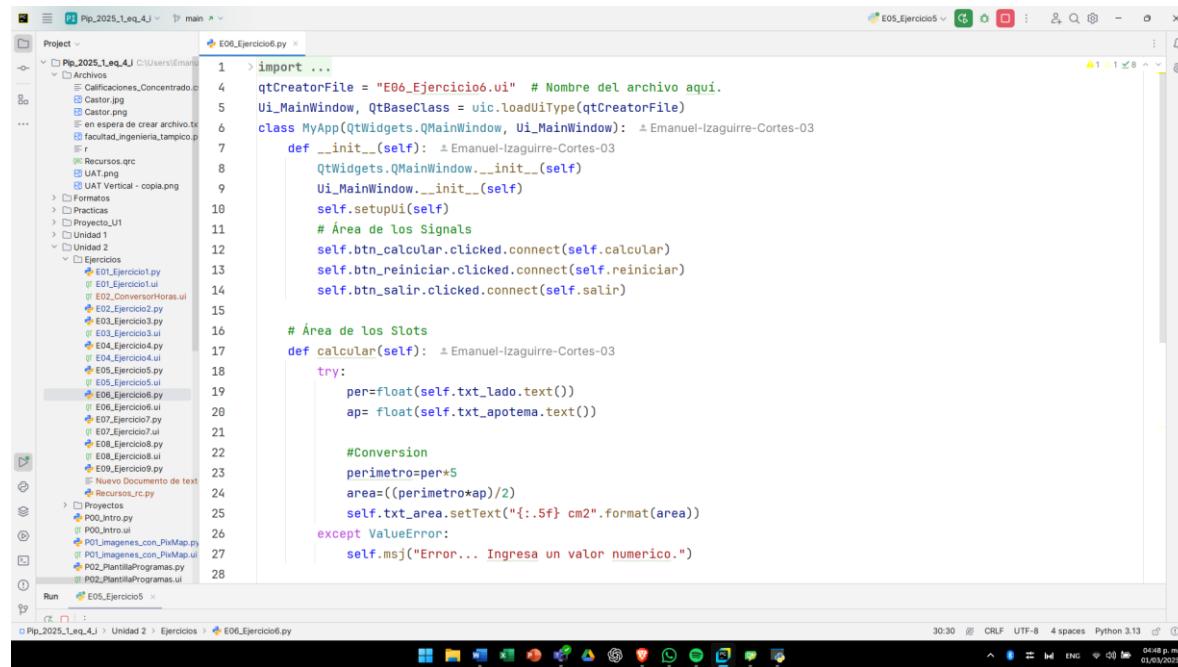
- Calcular:** Convierte el valor ingresado en pesos a dólares usando una tasa fija de 20.33 y muestra el resultado.
- Reiniciar:** Borra los valores de entrada y salida.
- Salir:** Cierra la aplicación.



Los elementos clave utilizados son:

- **PyQt5 y uic.loadUiType**: Para cargar la interfaz gráfica desde un archivo .ui sin necesidad de diseñarla manualmente en código.
- **Signals y Slots**: Para conectar botones a funciones (clicked.connect()).
- **Manejo de errores (try-except)**: Para evitar fallos si el usuario ingresa datos inválidos.
- **QMessageBox**: Para mostrar mensajes de error.

E06_ Área de un Pentágono

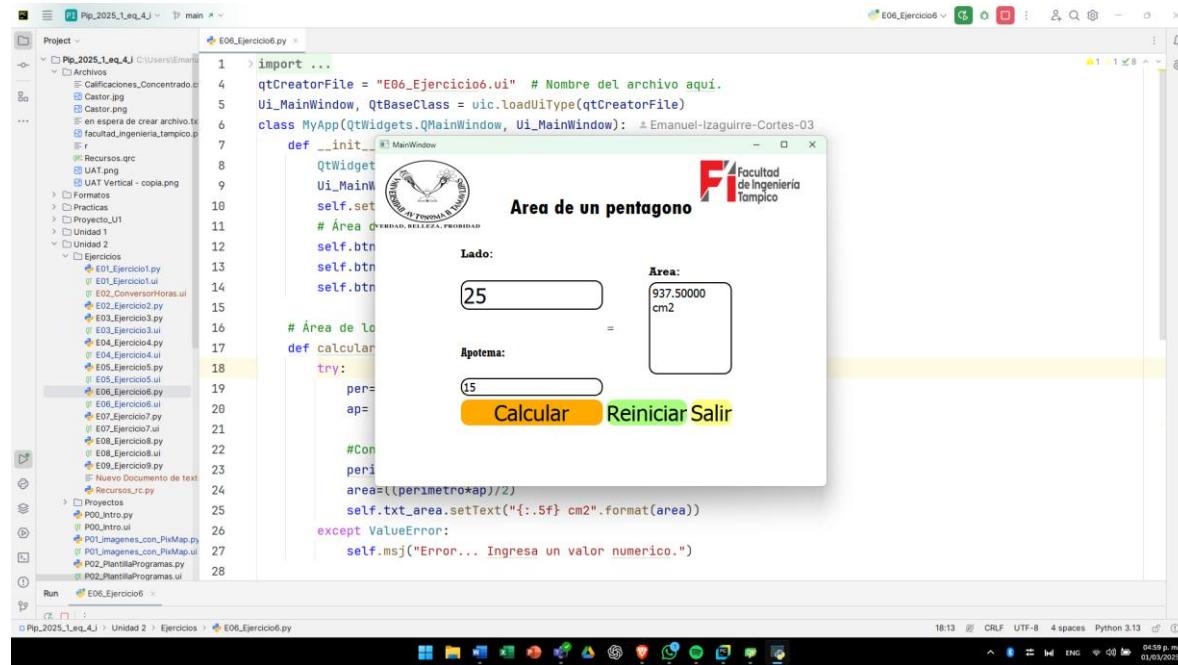


```
1 > import ...
4 qtCreatorFile = "E06_Ejercicio6.ui" # Nombre del archivo aquí.
5 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
6 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): # Emanuel-Izaguirre-Cortes-03
7     def __init__(self): # Emanuel-Izaguirre-Cortes-03
8         QtWidgets.QMainWindow.__init__(self)
9         Ui_MainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.btn_calcular.clicked.connect(self.calcular)
13        self.btn_reiniciar.clicked.connect(self.reiniciar)
14        self.btn_salir.clicked.connect(self.salir)
15
16        # Área de los Slots
17    def calcular(self): # Emanuel-Izaguirre-Cortes-03
18        try:
19            per=float(self.txt_lado.text())
20            ap= float(self.txt_apotema.text())
21
22            #Conversion
23            perimetro=per*5
24            area=((perimetro*ap)/2)
25            self.txt_area.setText("{:.5f} cm2".format(area))
26        except ValueError:
27            self.msg("Error... Ingresá un valor numérico.")
28
```



The screenshot shows a development environment with two main windows. The top window is a code editor for Python, displaying the file `E06_Ejercicio6.py`. The code defines a class `MyApp` that handles calculations for a pentagon based on side and apothem input. The bottom window is the Qt Designer interface, showing the graphical user interface for the application. The interface includes a logo for the Faculty of Engineering Tampico, a title bar "Area de un pentagono", and three input fields for "Lado", "Area", and "Ipotema". It features three buttons: "Calcular", "Reiniciar", and "Salir". The Qt Designer also displays toolbars, a central "Widget Box" containing various UI components, and several floating panels for "Object Inspector", "Property Editor", and "Resource Browser".

```
6     class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): □ Emanuel-Izaguirre-Cortes-03
7         def __init__(self): □ Emanuel-Izaguirre-Cortes-03
8             self.setWindowTitle("Área de un pentágono")
9             self.setCentralWidget(self.tabWidget)
10            self.tabWidget.setCurrentIndex(0)
11
12            self.retranslateUi(self)
13
14    15        def calcular(self): □ Emanuel-Izaguirre-Cortes-03
16            lado = float(self.txt_lado.text())
17            apotema = float(self.txt_apotema.text())
18            area = lado * apotema
19            self.txt_area.setText(str(area))
20
21        def reiniciar(self): □ Emanuel-Izaguirre-Cortes-03
22            self.txt_lado.clear()
23            self.txt_apotema.clear()
24            self.txt_area.clear()
25
26        def salir(self): □ Emanuel-Izaguirre-Cortes-03
27            self.close()
28
29        def msj(self, txt): □ Emanuel-Izaguirre-Cortes-03
30            m = QtWidgets.QMessageBox()
31            m.setText(txt)
32            m.exec_()
33
34    35    if __name__ == "__main__":
36        app = QtWidgets.QApplication(sys.argv)
37        window = MyApp()
38        window.show()
39        sys.exit(app.exec_())
40
41
42
43
44
45
46
```



Este programa es una aplicación de escritorio desarrollada en **Python con PyQt5** que calcula el **área de un pentágono regular** a partir del valor de un lado y su apotema.

Explicación de los elementos utilizados:

- 1. Carga de la interfaz gráfica (uic.loadUiType)**
Se usa un archivo .ui (E06_Ejercicio6.ui) diseñado con Qt Designer, lo que evita definir la interfaz manualmente en código.
- 2. Clase MyApp (Hereda de QMainWindow y la interfaz generada)**
 - Se configura la ventana principal con setupUi(self).
 - Se conectan los botones a funciones mediante **Signals y Slots**:
 - btn_calcular.clicked.connect(self.calcular) → Calcula el área.
 - btn_reiniciar.clicked.connect(self.reiniciar) → Limpia los campos.
 - btn_salir.clicked.connect(self.salir) → Cierra la aplicación.
- 3. Método calcular()**
 - Obtiene los valores ingresados de **lado** y **apotema** (txt_lado y txt_apotema).
 - Calcula el **perímetro** de un pentágono regular:
Perímetro=Lado×5\text{Perímetro} = \text{Lado} \times 5
 - Calcula el **área** usando la fórmula: $A' \text{rea} = \frac{\text{Perímetro} \times \text{Apotema}}{2}$
 - Muestra el resultado en txt_area, formateado con **5 decimales**.
- 4. Manejo de errores (try-except)**
 - Si el usuario ingresa un valor no numérico, se muestra un **QMessageBox** con un mensaje de error.

5. Métodos auxiliares

- `reiniciar()`: Limpia todos los campos de entrada y salida.
 - `salir()`: Cierra la aplicación.

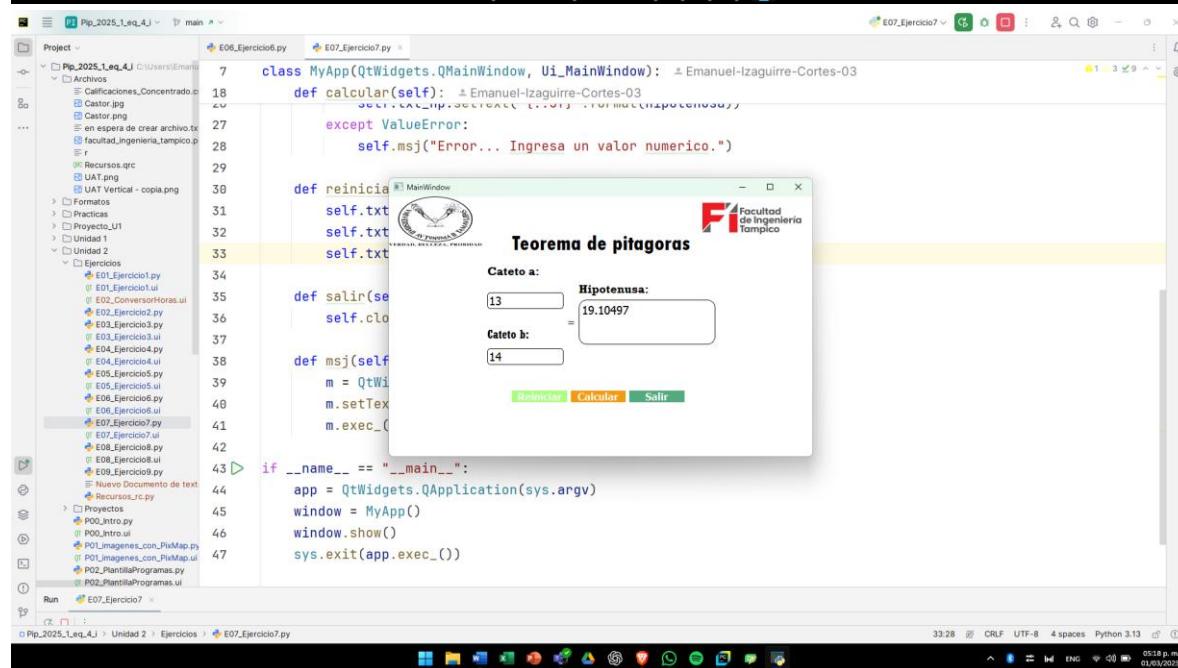
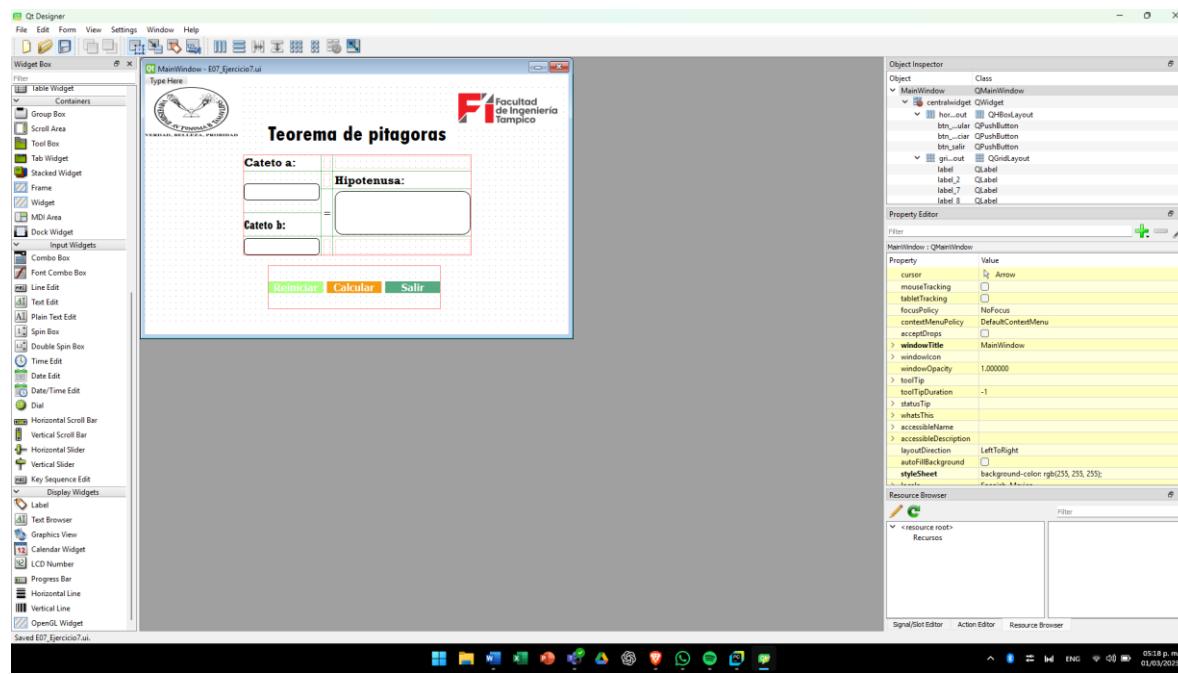
Conclusión

Este programa facilita el cálculo del área de un pentágono de forma rápida y sencilla. Su estructura modular y uso de PyQt5 permiten una **interfaz gráfica intuitiva y funcional**.

E07_ Teorema de Pitágoras

The screenshot shows the PyCharm IDE interface. The code editor displays Python code for a class `MyApp` that inherits from `QtWidgets.QMainWindow`. The code includes methods for calculating values, restarting, exiting, and displaying messages. The file browser on the left shows a project structure with multiple files and folders related to exercises and projects. The status bar at the bottom indicates the file is `E07_Ejercicio7.py` and provides other system information.

```
7     class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): □ Emanuel-Izaguirre-Cortes-03
18         def calcular(self): □ Emanuel-Izaguirre-Cortes-03
20             self.txt_hp.settext(str(result))
21         except ValueError:
22             self.msj("Error... Ingresa un valor numerico.")
23
24
25         def reiniciar(self): □ Emanuel-Izaguirre-Cortes-03
26             self.txt_a.clear()
27             self.txt_b.clear()
28             self.txt_hp.clear()
29
30         def salir(self): □ Emanuel-Izaguirre-Cortes-03
31             self.close()
32
33         def msj(self, txt): □ Emanuel-Izaguirre-Cortes-03
34             m = QtWidgets.QMessageBox()
35             m.setText(txt)
36             m.exec_()
37
38         if __name__ == "__main__":
39             app = QtWidgets.QApplication(sys.argv)
40             window = MyApp()
41             window.show()
42             sys.exit(app.exec_())
43
```





Esta aplicación es una calculadora sencilla del **Teorema de Pitágoras**. Permite ingresar las longitudes de los dos catetos (a y b) de un triángulo rectángulo y, al presionar el botón **Calcular**, se obtiene automáticamente la **hipotenusa** usando la fórmula:

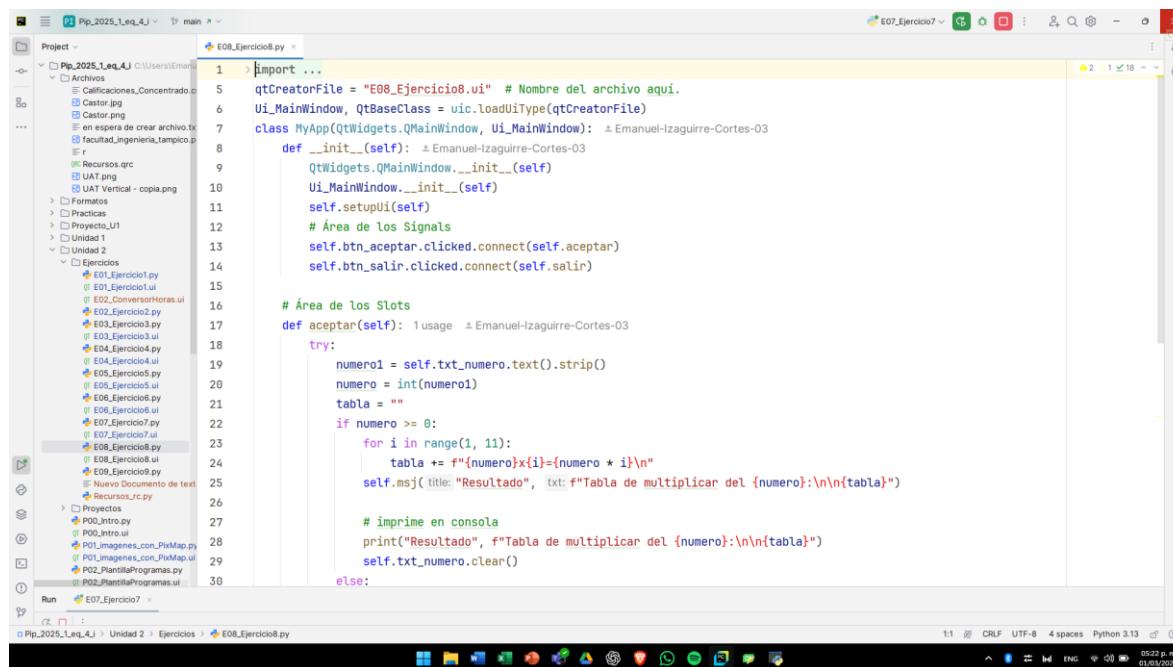
$$\text{Hipotenusa} = \sqrt{(\text{Cateto } a)^2 + (\text{Cateto } b)^2}$$

También incluye los botones:

- **Reiniciar**: Limpia los campos de entrada y el resultado.
- **Salir**: Cierra la aplicación.

De esta forma, el programa facilita el cálculo rápido y preciso de la hipotenusa de un triángulo rectángulo.

E08_ Tabla de multiplicar



```
1 #import ...
5 qtcreatorfile = "E08_Ejercicio8.ui" # Nombre del archivo aqui.
6 Ui_MainWindow, QtBaseClass = uic.loadUiType(qtcreatorFile)
7 class MyApp(QtWidgets.QMainWindow, Ui_MainWindow): # Emanuel-Izaguirre-Cortes-03
8     def __init__(self): # Emanuel-Izaguirre-Cortes-03
9         QtWidgets.QMainWindow.__init__(self)
10        self.setupUi(self)
11        # Área de los Signals
12        self.btn_aceptar.clicked.connect(self.aceptar)
13        self.btn_salir.clicked.connect(self.salir)
14
15 # Área de los Slots
16 def aceptar(self): # usage # Emanuel-Izaguirre-Cortes-03
17     try:
18         numero1 = self.txt_numero.text().strip()
19         numero = int(numero1)
20         tabla = ""
21         if numero >= 0:
22             for i in range(1, 11):
23                 tabla += f"{numero}{x}{i}={numero * i}\n"
24             self.msg( title: "Resultado", txt: f"Tabla de multiplicar del {numero}:\n\n{tabla}")
25
26             # imprime en consola
27             print("Resultado", f"Tabla de multiplicar del {numero}:\n\n{tabla}")
28             self.txt_numero.clear()
29
30     else:
```

A screenshot of a Windows desktop environment showing a Qt application development setup. The top half of the screen displays a code editor with Python code for an application named 'E08_Ejercicio8.py'. The code handles user input and displays messages using QMessageBox. The bottom half shows the resulting application window titled 'E08_Ejercicio8.ui'. The window features a logo for 'Facultad de Ingeniería Tampico' and contains a label 'Tabla de multiplicar' and an instruction 'Ingresa un numero:'. It has two buttons at the bottom: 'Aceptar' and 'Salir'. The Qt Designer interface is visible on the left, showing various widget components like Table Widget, Container, Group Box, etc. On the right, there are Object Inspector, Property Editor, and Resource Browser panels. The taskbar at the bottom shows other open applications like Microsoft Word and Excel.



The screenshot shows the PyCharm IDE interface. On the left is the project tree for 'Pip_2025_1_leq_4_J'. The main editor window displays the Python script 'E08_Ejercicio8.py'. The code implements a multiplication table generator. A small application window titled 'Tabla de multiplicar' is overlaid on the IDE, prompting the user to 'Ingresar un número' (Enter a number). The user has entered '7' and clicked 'Aceptar' (Accept). The application then displays the multiplication table for 7 from 1 to 10. The table output is:

7x1=7
7x2=14
7x3=21
7x4=28
7x5=35
7x6=42
7x7=49
7x8=56
7x9=63
7x10=70

Esta aplicación muestra la **tabla de multiplicar** de un número que el usuario ingresa. Al dar clic en el botón **Aceptar**, se calcula y se despliega en un cuadro de diálogo la multiplicación de ese número por los valores del 1 al 10. También se imprime el mismo resultado en la consola. Finalmente, el botón **Salir** cierra la aplicación. De esta manera, el programa permite al usuario obtener de forma rápida y sencilla la tabla de multiplicar del número que deseé.