Actividad 08 – QTableWidget

Hernández Lomelí Diego Armando

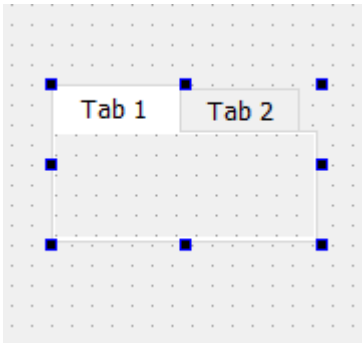
Seminario de algoritmia 2022B D02

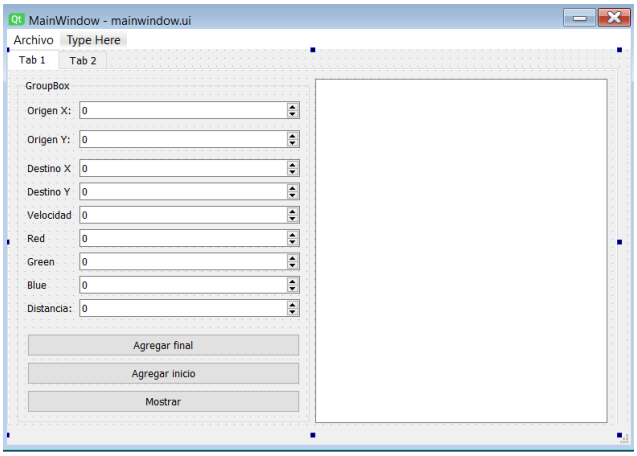
Lineamientos de evaluación.

- [ ] El reporte está en formato Google Docs o PDF.
- [ ] El reporte sigue las pautas del Formato de Actividades .
- [ ] El reporte tiene desarrollada todas las pautas del Formato de Actividades.
- [ ] Se muestra captura de pantalla de lo que se pide en el punto 2. sub punto a.
- [ ] Se muestra captura de pantalla de lo que se pide en el punto 2. sub punto b.
- [ ] Se muestra captura de pantalla de lo que se pide en el punto 2. sub punto c.
- [ ] Se muestra captura de pantalla de lo que se pide en el punto 2. sub punto d.

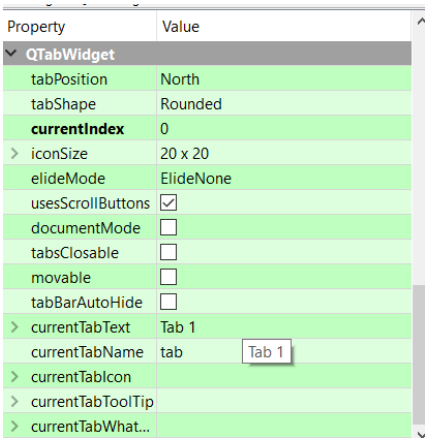
Desarrollo

Debemos agrandar el componente y meter dentro del todos los componentes incluidos en trabajos anteriores dentro de **Tab1**, solo falta arrastrarlos componentes.



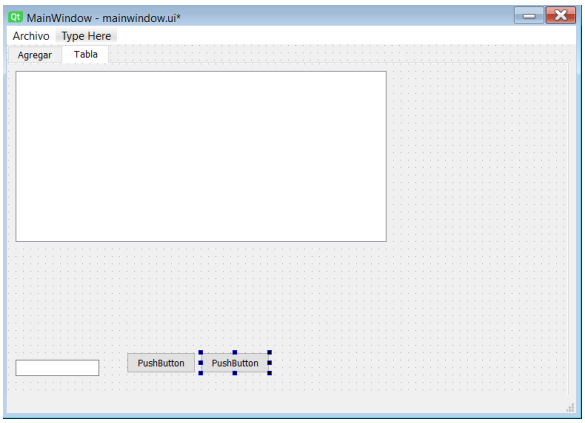


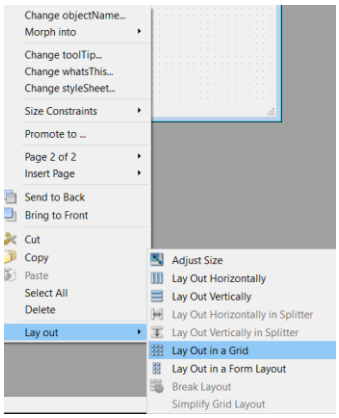
En las propiedades editamos **currentTabText** para modificar el texto de la tabulación.



Incluimos el **TableWidget** en la segunda tabulación, para acceder a su espacio solamente debemos dar clic en su etiqueta

Lo agregamos y justo debajo también ponemos un **LineEdit** y 2 botones.

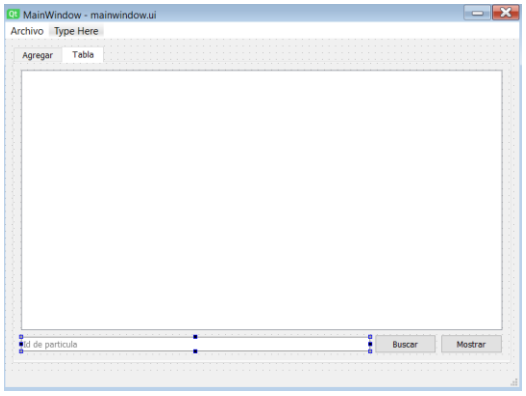




Después elegimos con click derecho en el espacio que estamos usando y le agregamos el **layout** "**lay out in a grid"**

El siguiente paso es cambiar etiquetas de texto y cambiar el nombre de los componentes individuales.

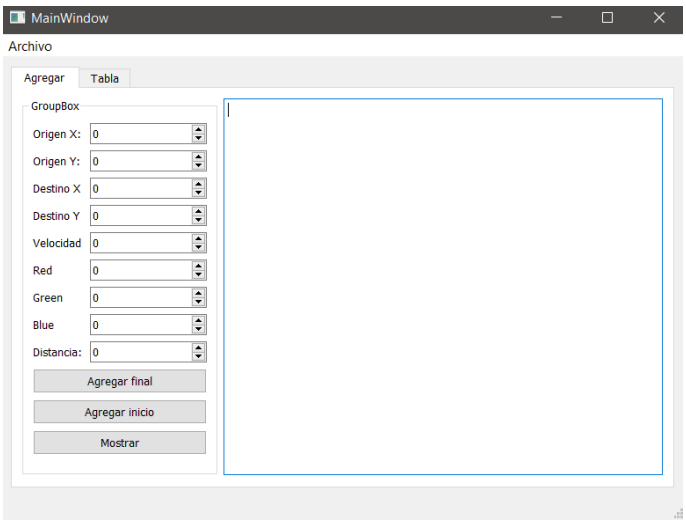
Este es el resultado.



Estos son todos los cambios que vamos a realizar en la ventana. Seguimos desde Python y para eso primeramente convertimos el archivo **.ui** a uno **.py.**

```
PS C:\Users\Armando\Documents\GitHub\Actividad 8> C:\Users\Armando\AppData\Local\Programs\Python\Python310\Scripts\pyside2-uic "C:\Users\Armando\Documents\GitHub\Actividad 8\src\mainwindow.ui" -o ui_main
window.py
```

Y ya podemos ejecutar la ventana desde Python.



Toca agregar eventos a los botones que creamos anteriormente, empezaremos con el btn para mostrar, con el tendremos una vista a las partículas que hemos almacenado.

```
""" Metodos para trabajar con la tabla """
self.ui.mostrar_pushButton.clicked.connect(self.mostrar_tabla)
```

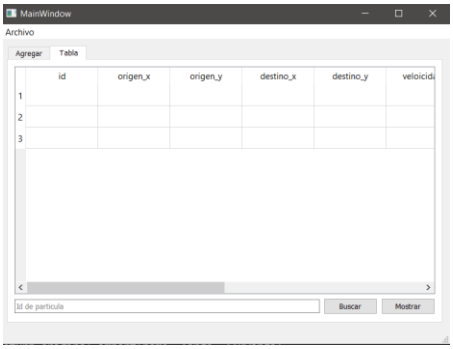
Este método nos genera las columnas de la tabla pero debemos implementar el método **len** a la lista de partículas para evitar errores en la compilación.

```
@Slot
def mostrar_tabla(self):
    self.ui.tableParticulas.setColumnCount(10)
    headers = ["id",
               "origen_x",
               "origen_y",
               "destino_x",
               "destino_y",      You, now • Uncommitted changes
               "veloicidad",
               "red",
               "green",
               "blue",
               "distancia"]
    self.ui.tableParticulas.setHorizontalHeaderLabels(headers)
    self.ui.tableParticulas.setRowCount(len(self.__lista))
```

El siguiente método debe estar dentro de la clase **listaParticula**.

```
def __len__(self):
    return len(self.__particulas)
```

Ahora podemos probar la funcionalidad del botón. Al presionar el botón para buscar, veremos que la tabla ha cambiado y ha agregado filas para la cantidad de partículas que respaldamos en la actividad anterior (3 en total)



Debemos hacer que los objetos de la lista de partículas sean visibles desde acá, para ello debemos hacer que la instancia de **listaParticulas** sea iterable, esto es con propósito a no acceder directamente a la lista.

Dentro de la clase **listaParticula** agregamos la siguiente función.

```
def __iter__(self):
    self.cont = 0
    return self
```

La siguiente función estará dentro de la misma clase y nos devolverá el objeto que debería seguir en la iteración, esto sin acceder directamente a la lista **partículas**.

```
def __next__(self):
    if self.cont < len(self.__particulas):
        """ Asignamos la particula a devolver """
        particula = self.__particulas[self.cont]
        """ Incremenamos el contador """
        self.cont += 1
        return particula
    """ detemos la iteración si se sobrepasa el tamaño de la lista """
    raise StopIteration
```

Antes de volver a **mainwindow.py** debemos agregar modificadores de acceso a la clase **particula** para poder leerlos desde otra clase.

Estos getters tendrán la siguiente forma:

```
@property
def id(self):
    return self.__id

@property
def origen_x(self):
    return self.__origen_x
```

Debemos repetir para cada campo que quedamos acceder desde una clase externa

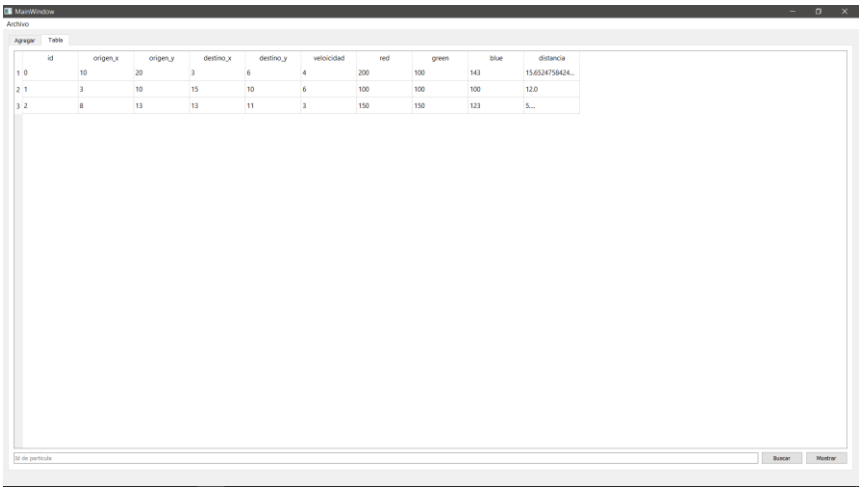
Seguimos con la importación de la clase **QTableWidgetItem** en la clase **mainwindow.py** para empezar a rellenar de datos la tabla.

```python
""" Empezamos a rellenar la tabla """
row = 0
for particula in self.__lista:
    id_widget = QTableWidgetItem(str(particula.id))
    origen_x_widget = QTableWidgetItem(str(particula.origen_x))
    origen_y_widget = QTableWidgetItem(str(particula.origen_y))
    destino_x_widget = QTableWidgetItem(str(particula.destino_x))
    destino_y_widget = QTableWidgetItem(str(particula.destino_y))
    veloicidad_widget = QTableWidgetItem(str(particula.veloicidad))
    red_widget = QTableWidgetItem(str(particula.red))
    green_widget = QTableWidgetItem(str(particula.green))
    blue_widget = QTableWidgetItem(str(particula.blue))
    distancia_widget = QTableWidgetItem(str(particula.distancia))

    self.ui.tableParticulas.setItem(row, 0, id_widget)
    self.ui.tableParticulas.setItem(row, 1, origen_x_widget)
    self.ui.tableParticulas.setItem(row, 2, origen_y_widget)
    self.ui.tableParticulas.setItem(row, 3, destino_x_widget)
    self.ui.tableParticulas.setItem(row, 4, destino_y_widget)
    self.ui.tableParticulas.setItem(row, 5, veloicidad_widget)
    self.ui.tableParticulas.setItem(row, 6, red_widget)
    self.ui.tableParticulas.setItem(row, 7, green_widget)
    self.ui.tableParticulas.setItem(row, 8, blue_widget)
    self.ui.tableParticulas.setItem(row, 9, distancia_widget)

    row += 1
```

Retomamos el método **mostrar_tabla** desde donde lo dejamos y agregamos el código de la imagen, con cada **QTableWidgetItem** estamos generando una nueva celda.

Cuando usamos el método **setItem** de **QTable** decidimos en que fila y en que columna queremos agregar el objeto.

Si probamos el método después de haber cargado datos previamente guardados, tendremos el siguiente resultado, ya son visibles desde la lista.



Lo que procede ahora es incluir el método de búsqueda mediante el id de la partícula

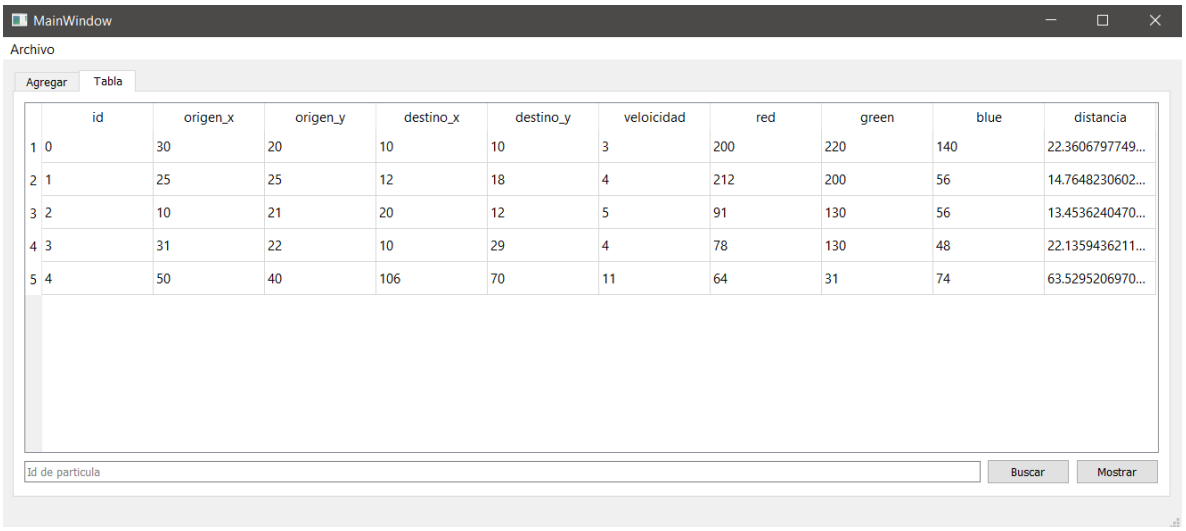
```python
@Slot()
def buscar_id_tabla(self):
    idBusqueda = self.ui.searchEdit.text()

    for particula in self.__lista:      You, 2 minutes ago • Uncommitted changes
        if idBusqueda == str(particula.id):
            self.ui.tableParticulas.clear()
            self.ui.tableParticulas.setColumnCount(10)
            headers = ["id", "origen_x", "origen_y", "destino_x",
                       "destino_y", "veloicidad", "red", "green", "blue", "distancia"]
            self.ui.tableParticulas.setHorizontalHeaderLabels(headers)
            self.ui.tableParticulas.setRowCount(1)

            id_widget = QTableWidgetItem(str(particula.id))
            origen_x_widget = QTableWidgetItem(str(particula.origen_x))
            origen_y_widget = QTableWidgetItem(str(particula.origen_y))
            destino_x_widget = QTableWidgetItem(str(particula.destino_x))
            destino_y_widget = QTableWidgetItem(str(particula.destino_y))
            veloicidad_widget = QTableWidgetItem(str(particula.veloicidad))
            red_widget = QTableWidgetItem(str(particula.red))
            green_widget = QTableWidgetItem(str(particula.green))
            blue_widget = QTableWidgetItem(str(particula.blue))
            distancia_widget = QTableWidgetItem(str(particula.distancia))

            self.ui.tableParticulas.setItem(0, 0, id_widget)
            self.ui.tableParticulas.setItem(0, 1, origen_x_widget)
            self.ui.tableParticulas.setItem(0, 2, origen_y_widget)
            self.ui.tableParticulas.setItem(0, 3, destino_x_widget)
            self.ui.tableParticulas.setItem(0, 4, destino_y_widget)
            self.ui.tableParticulas.setItem(0, 5, veloicidad_widget)
            self.ui.tableParticulas.setItem(0, 6, red_widget)
            self.ui.tableParticulas.setItem(0, 7, green_widget)
            self.ui.tableParticulas.setItem(0, 8, blue_widget)
            self.ui.tableParticulas.setItem(0, 9, distancia_widget)

            return
    QMessageBox.warning(
        self,
        "Exito",
        f'No se ha encontrado una particula con el id: "{idBusqueda}"'
    )
```

Parece ser muy extenso, pero por la cantidad de atributos toma mínimo 18 líneas solo para agregar una fila a la tabla. La búsqueda es línea entre los elementos del arreglo.

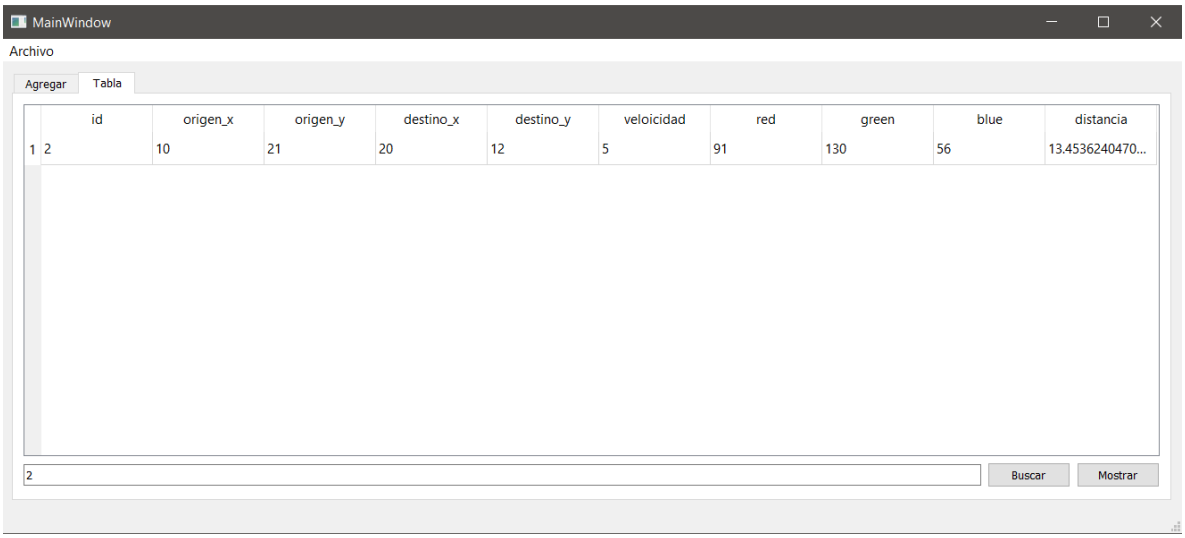
Ya podemos empezar a hacer una mejor prueba, para ello generaré un respaldo de 5 partículas y lo cargaré en la ventana, después haré una búsqueda con intención a ser exitosa y luego otra con intención a ser fallida.

a) Agregar o recuperar un respaldo de por lo menos 5 particulas.

b) Mostrar las partículas en el **QTableWidget.**

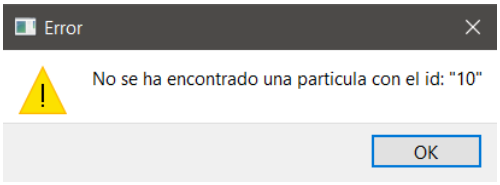


c) Realizar la búsqueda de una partícula con un id existente.
Para este ejemplo buscaremos la partícula con el id **“2”.**



Solo muestra la partícula que coincide en la búsqueda, todas las demás no son vsibles.

d) Realizar la búsqueda de una partícula con un id no existente.
Para este ejemplo buscaremos las partícula con el id **“10”.**



Conclusiones

En esta actividad logre desarrollar los requerimientos con ayuda del material proporcionado, lo que me sorprendió en esta actividad fue la escritura de métodos que solo puedo suponer que son sobre escritos para poder trabajar con ellos (método len y next) por las características del lenguaje, sobre el manejo de la tabla me recuerda al manejo que se le da desde javax.swing.JTable, pero se usan más instancias diferentes para hacer que se desplieguen sus datos. En general fue una buena actividad para mejorar practicas en el acceso a atributos de una clase dentro de otra.

Referencia

BOITES, M. D. (29 de Octubre de 2020). Obtenido de Youtube:
https://www.youtube.com/watch?v=1yEpAHaiMxs

Código

Ui_mainwindow.py

```
# -*- coding: utf-8 -*-

################################################################################
## Form generated from reading UI file 'mainwindow.ui'
##
## Created by: Qt User Interface Compiler version 5.15.2
##
## WARNING! All changes made in this file will be lost when recompiling UI file!
################################################################################

from PySide2.QtCore import *
from PySide2.QtGui import *
from PySide2.QtWidgets import *


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        if not MainWindow.objectName():
            MainWindow.setObjectName(u"MainWindow")
        MainWindow.resize(779, 552)
        self.actionAbrir = QAction(MainWindow)
        self.actionAbrir.setObjectName(u"actionAbrir")
        self.actionGuardar = QAction(MainWindow)
        self.actionGuardar.setObjectName(u"actionGuardar")
        self.centralwidget = QWidget(MainWindow)
        self.centralwidget.setObjectName(u"centralwidget")
        self.gridLayout_3 = QGridLayout(self.centralwidget)
        self.gridLayout_3.setObjectName(u"gridLayout_3")
        self.tabWidget = QTabWidget(self.centralwidget)
        self.tabWidget.setObjectName(u"tabWidget")
        self.tab = QWidget()
        self.tab.setObjectName(u"tab")
        self.gridLayout = QGridLayout(self.tab)
        self.gridLayout.setObjectName(u"gridLayout")
        self.groupBox = QGroupBox(self.tab)
        self.groupBox.setObjectName(u"groupBox")
        self.formLayout = QFormLayout(self.groupBox)
        self.formLayout.setObjectName(u"formLayout")
        self.label = QLabel(self.groupBox)
        self.label.setObjectName(u"label")

        self.formLayout.setWidget(4, QFormLayout.LabelRole, self.label)
```

```
        self.spnnDestinoX = QSpinBox(self.groupBox)
        self.spnnDestinoX.setObjectName(u"spnnDestinoX")
        self.spnnDestinoX.setMaximum(500)

        self.formLayout.setWidget(4, QFormLayout.FieldRole, self.spnnDestinoX)

        self.label_2 = QLabel(self.groupBox)
        self.label_2.setObjectName(u"label_2")

        self.formLayout.setWidget(5, QFormLayout.LabelRole, self.label_2)

        self.spnnDestinoY = QSpinBox(self.groupBox)
        self.spnnDestinoY.setObjectName(u"spnnDestinoY")
        self.spnnDestinoY.setMaximum(500)

        self.formLayout.setWidget(5, QFormLayout.FieldRole, self.spnnDestinoY)

        self.label_3 = QLabel(self.groupBox)
        self.label_3.setObjectName(u"label_3")

        self.formLayout.setWidget(6, QFormLayout.LabelRole, self.label_3)

        self.label_5 = QLabel(self.groupBox)
        self.label_5.setObjectName(u"label_5")

        self.formLayout.setWidget(7, QFormLayout.LabelRole, self.label_5)

        self.spnnRed = QSpinBox(self.groupBox)
        self.spnnRed.setObjectName(u"spnnRed")
        self.spnnRed.setMaximum(255)

        self.formLayout.setWidget(7, QFormLayout.FieldRole, self.spnnRed)

        self.label_6 = QLabel(self.groupBox)
        self.label_6.setObjectName(u"label_6")

        self.formLayout.setWidget(8, QFormLayout.LabelRole, self.label_6)

        self.spnnGreen = QSpinBox(self.groupBox)
        self.spnnGreen.setObjectName(u"spnnGreen")
        self.spnnGreen.setMaximum(255)

        self.formLayout.setWidget(8, QFormLayout.FieldRole, self.spnnGreen)
```

```
        self.label_7 = QLabel(self.groupBox)
        self.label_7.setObjectName(u"label_7")

        self.formLayout.setWidget(9, QFormLayout.LabelRole, self.label_7)

        self.spnnBlue = QSpinBox(self.groupBox)
        self.spnnBlue.setObjectName(u"spnnBlue")
        self.spnnBlue.setMaximum(255)

        self.formLayout.setWidget(9, QFormLayout.FieldRole, self.spnnBlue)

        self.btnAgregarFinal = QPushButton(self.groupBox)
        self.btnAgregarFinal.setObjectName(u"btnAgregarFinal")
        self.btnAgregarFinal.setCursor(QCursor(Qt.PointingHandCursor))

        self.formLayout.setWidget(13, QFormLayout.SpanningRole, self.btnAgregarFinal)

        self.label_4 = QLabel(self.groupBox)
        self.label_4.setObjectName(u"label_4")

        self.formLayout.setWidget(0, QFormLayout.LabelRole, self.label_4)

        self.spnnOrigenX = QSpinBox(self.groupBox)
        self.spnnOrigenX.setObjectName(u"spnnOrigenX")
        self.spnnOrigenX.setMaximum(500)

        self.formLayout.setWidget(0, QFormLayout.FieldRole, self.spnnOrigenX)

        self.spnnOrigenY = QSpinBox(self.groupBox)
        self.spnnOrigenY.setObjectName(u"spnnOrigenY")
        self.spnnOrigenY.setMaximum(500)

        self.formLayout.setWidget(2, QFormLayout.FieldRole, self.spnnOrigenY)

        self.label_8 = QLabel(self.groupBox)
        self.label_8.setObjectName(u"label_8")

        self.formLayout.setWidget(2, QFormLayout.LabelRole, self.label_8)

        self.spnnVelocidad = QSpinBox(self.groupBox)
        self.spnnVelocidad.setObjectName(u"spnnVelocidad")
        self.spnnVelocidad.setMaximum(500)

        self.formLayout.setWidget(6, QFormLayout.FieldRole, self.spnnVelocidad)
```

```
        self.btnAgregarInicio = QPushButton(self.groupBox)
        self.btnAgregarInicio.setObjectName(u"btnAgregarInicio")
        self.btnAgregarInicio.setCursor(QCursor(Qt.PointingHandCursor))

        self.formLayout.setWidget(14, QFormLayout.SpanningRole, self.btnAgregarInicio)

        self.btnMostrar = QPushButton(self.groupBox)
        self.btnMostrar.setObjectName(u"btnMostrar")
        self.btnMostrar.setCursor(QCursor(Qt.PointingHandCursor))

        self.formLayout.setWidget(15, QFormLayout.SpanningRole, self.btnMostrar)

        self.label_9 = QLabel(self.groupBox)
        self.label_9.setObjectName(u"label_9")

        self.formLayout.setWidget(10, QFormLayout.LabelRole, self.label_9)

        self.spnnDistancia = QSpinBox(self.groupBox)
        self.spnnDistancia.setObjectName(u"spnnDistancia")
        self.spnnDistancia.setMaximum(255)

        self.formLayout.setWidget(10, QFormLayout.FieldRole, self.spnnDistancia)


        self.gridLayout.addWidget(self.groupBox, 0, 0, 1, 1)

        self.plainTextEdit = QPlainTextEdit(self.tab)
        self.plainTextEdit.setObjectName(u"plainTextEdit")
        self.plainTextEdit.setMaximumSize(QSize(500, 16777215))
        self.plainTextEdit.setFrameShadow(QFrame.Raised)

        self.gridLayout.addWidget(self.plainTextEdit, 0, 1, 1, 1)

        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QWidget()
        self.tab_2.setObjectName(u"tab_2")
        self.gridLayout_2 = QGridLayout(self.tab_2)
        self.gridLayout_2.setObjectName(u"gridLayout_2")
        self.tableParticulas = QTableWidget(self.tab_2)
        self.tableParticulas.setObjectName(u"tableParticulas")

        self.gridLayout_2.addWidget(self.tableParticulas, 0, 0, 1, 3)

        self.searchEdit = QLineEdit(self.tab_2)
        self.searchEdit.setObjectName(u"searchEdit")
```

```
        self.gridLayout_2.addWidget(self.searchEdit, 1, 0, 1, 1)

        self.buscar_pushButton = QPushButton(self.tab_2)
        self.buscar_pushButton.setObjectName(u"buscar_pushButton")

        self.gridLayout_2.addWidget(self.buscar_pushButton, 1, 1, 1, 1)

        self.mostrar_pushButton = QPushButton(self.tab_2)
        self.mostrar_pushButton.setObjectName(u"mostrar_pushButton")

        self.gridLayout_2.addWidget(self.mostrar_pushButton, 1, 2, 1, 1)

        self.tabWidget.addTab(self.tab_2, "")

        self.gridLayout_3.addWidget(self.tabWidget, 0, 0, 1, 1)

        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QMenuBar(MainWindow)
        self.menubar.setObjectName(u"menubar")
        self.menubar.setGeometry(QRect(0, 0, 779, 26))
        self.menuArchivo = QMenu(self.menubar)
        self.menuArchivo.setObjectName(u"menuArchivo")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QStatusBar(MainWindow)
        self.statusbar.setObjectName(u"statusbar")
        MainWindow.setStatusBar(self.statusbar)

        self.menubar.addAction(self.menuArchivo.menuAction())
        self.menuArchivo.addAction(self.actionAbrir)
        self.menuArchivo.addAction(self.actionGuardar)

        self.retranslateUi(MainWindow)

        self.tabWidget.setCurrentIndex(0)


        QMetaObject.connectSlotsByName(MainWindow)
    # setupUi

    def retranslateUi(self, MainWindow):
        MainWindow.setWindowTitle(QCoreApplication.translate("MainWindow",
u"MainWindow", None))
        self.actionAbrir.setText(QCoreApplication.translate("MainWindow", u"Abrir",
None))
```

```
#if QT_CONFIG(shortcut)
        self.actionAbrir.setShortcut(QCoreApplication.translate("MainWindow",
u"Ctrl+O", None))
#endif // QT_CONFIG(shortcut)
        self.actionGuardar.setText(QCoreApplication.translate("MainWindow", u"Guardar",
None))
#if QT_CONFIG(shortcut)
        self.actionGuardar.setShortcut(QCoreApplication.translate("MainWindow",
u"Ctrl+G", None))
#endif // QT_CONFIG(shortcut)
        self.groupBox.setTitle(QCoreApplication.translate("MainWindow", u"GroupBox",
None))
        self.label.setText(QCoreApplication.translate("MainWindow", u"Destino X",
None))
        self.label_2.setText(QCoreApplication.translate("MainWindow", u"Destino Y",
None))
        self.label_3.setText(QCoreApplication.translate("MainWindow", u"Velocidad",
None))
        self.label_5.setText(QCoreApplication.translate("MainWindow", u"Red", None))
        self.label_6.setText(QCoreApplication.translate("MainWindow", u"Green", None))
        self.label_7.setText(QCoreApplication.translate("MainWindow", u"Blue", None))
        self.btnAgregarFinal.setText(QCoreApplication.translate("MainWindow", u"Agregar
final", None))
        self.label_4.setText(QCoreApplication.translate("MainWindow", u"Origen X:",
None))
        self.label_8.setText(QCoreApplication.translate("MainWindow", u"Origen Y: ",
None))
        self.btnAgregarInicio.setText(QCoreApplication.translate("MainWindow",
u"Agregar inicio", None))
        self.btnMostrar.setText(QCoreApplication.translate("MainWindow", u"Mostrar",
None))
        self.label_9.setText(QCoreApplication.translate("MainWindow", u"Distancia:",
None))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab),
QCoreApplication.translate("MainWindow", u"Agregar", None))
        self.searchEdit.setPlaceholderText(QCoreApplication.translate("MainWindow",
u"Id de particula", None))
        self.buscar_pushButton.setText(QCoreApplication.translate("MainWindow",
u"Buscar", None))
        self.mostrar_pushButton.setText(QCoreApplication.translate("MainWindow",
u"Mostrar", None))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2),
QCoreApplication.translate("MainWindow", u"Tabla", None))
        self.menuArchivo.setTitle(QCoreApplication.translate("MainWindow", u"Archivo",
None))
```

```
    # retranslateUi
```

mainwindow.py

```
from ui_mainwindow import Ui_MainWindow, QFileDialog, QMessageBox, QTableWidgetItem
from PySide2.QtWidgets import QMainWindow
from PySide2.QtCore import Slot
from listaParticulas import listaParticula
from Particula import Particula


class MainWindow(QMainWindow):
    __contador = 0

    def __init__(self):
        super(MainWindow, self).__init__()
        self.__lista = listaParticula()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.ui.btnAgregarInicio.clicked.connect(self.click_agregar_inicio)
        self.ui.btnAgregarFinal.clicked.connect(self.click_agregar_final)
        self.ui.btnMostrar.clicked.connect(self.mostrar)
        """ Metodos para el menu de la ventana """
        self.ui.actionAbrir.triggered.connect(self.action_abrir_archivo)
        self.ui.actionGuardar.triggered.connect(self.action_guardar_archivo)
        """ Metodos para trabajar con la tabla """
        self.ui.mostrar_pushButton.clicked.connect(self.mostrar_tabla)
        self.ui.buscar_pushButton.clicked.connect(self.buscar_id_tabla)

    @Slot()
    def mostrar_tabla(self):
        self.ui.tableParticulas.setColumnCount(10)
        headers = ["id", "origen_x", "origen_y", "destino_x",
                   "destino_y", "veloicidad", "red", "green", "blue", "distancia"]
        self.ui.tableParticulas.setHorizontalHeaderLabels(headers)
        self.ui.tableParticulas.setRowCount(len(self.__lista))
        """ Empezamos a rellenar la tabla """
        row = 0
        for particula in self.__lista:
            id_widget = QTableWidgetItem(str(particula.id))
            origen_x_widget = QTableWidgetItem(str(particula.origen_x))
            origen_y_widget = QTableWidgetItem(str(particula.origen_y))
            destino_x_widget = QTableWidgetItem(str(particula.destino_x))
            destino_y_widget = QTableWidgetItem(str(particula.destino_y))
            veloicidad_widget = QTableWidgetItem(str(particula.veloicidad))
```

```
            red_widget = QTableWidgetItem(str(particula.red))
            green_widget = QTableWidgetItem(str(particula.green))
            blue_widget = QTableWidgetItem(str(particula.blue))
            distancia_widget = QTableWidgetItem(str(particula.distancia))

            self.ui.tableParticulas.setItem(row, 0, id_widget)
            self.ui.tableParticulas.setItem(row, 1, origen_x_widget)
            self.ui.tableParticulas.setItem(row, 2, origen_y_widget)
            self.ui.tableParticulas.setItem(row, 3, destino_x_widget)
            self.ui.tableParticulas.setItem(row, 4, destino_y_widget)
            self.ui.tableParticulas.setItem(row, 5, veloicidad_widget)
            self.ui.tableParticulas.setItem(row, 6, red_widget)
            self.ui.tableParticulas.setItem(row, 7, green_widget)
            self.ui.tableParticulas.setItem(row, 8, blue_widget)
            self.ui.tableParticulas.setItem(row, 9, distancia_widget)

            row += 1

    @Slot()
    def buscar_id_tabla(self):
        idBusqueda = self.ui.searchEdit.text()

        for particula in self.__lista:
            if idBusqueda == str(particula.id):
                self.ui.tableParticulas.clear()
                self.ui.tableParticulas.setColumnCount(10)
                headers = ["id", "origen_x", "origen_y", "destino_x",
                           "destino_y", "veloicidad", "red", "green", "blue",
"distancia"]
                self.ui.tableParticulas.setHorizontalHeaderLabels(headers)
                self.ui.tableParticulas.setRowCount(1)

                id_widget = QTableWidgetItem(str(particula.id))
                origen_x_widget = QTableWidgetItem(str(particula.origen_x))
                origen_y_widget = QTableWidgetItem(str(particula.origen_y))
                destino_x_widget = QTableWidgetItem(str(particula.destino_x))
                destino_y_widget = QTableWidgetItem(str(particula.destino_y))
                veloicidad_widget = QTableWidgetItem(str(particula.veloicidad))
                red_widget = QTableWidgetItem(str(particula.red))
                green_widget = QTableWidgetItem(str(particula.green))
                blue_widget = QTableWidgetItem(str(particula.blue))
                distancia_widget = QTableWidgetItem(str(particula.distancia))

                self.ui.tableParticulas.setItem(0, 0, id_widget)
                self.ui.tableParticulas.setItem(0, 1, origen_x_widget)
```

```
            self.ui.tableParticulas.setItem(0, 2, origen_y_widget)
            self.ui.tableParticulas.setItem(0, 3, destino_x_widget)
            self.ui.tableParticulas.setItem(0, 4, destino_y_widget)
            self.ui.tableParticulas.setItem(0, 5, veloicidad_widget)
            self.ui.tableParticulas.setItem(0, 6, red_widget)
            self.ui.tableParticulas.setItem(0, 7, green_widget)
            self.ui.tableParticulas.setItem(0, 8, blue_widget)
            self.ui.tableParticulas.setItem(0, 9, distancia_widget)

            return
    QMessageBox.warning(
        self,
        "Error",
        f'No se ha encontrado una particula con el id: "{idBusqueda}"'
    )

""" Generación de eventos para acciones del menu """
@Slot()
def action_guardar_archivo(self):
    ubicacion = QFileDialog.getSaveFileName(
        self,
        "Guardar archivo",
        ".",
        "JSON (*.json)"
    )[0]
    if self.__lista.guardar(ubicacion):
        QMessageBox.information(
            self,
            "Exito",
            ("Se pudo crear el archivo " + ubicacion)
        )
    else:
        QMessageBox.critical(
            self,
            "Error",
            ("No pudo crear el archivo " + ubicacion)
        )

@Slot()
def action_abrir_archivo(self):
    ubicacion = QFileDialog.getSaveFileName(
        self,
        "Guardar archivo",
        ".",
        "JSON (*.json)"
```

```
        )[0]
        if self.__lista.abrir(ubicacion):
            QMessageBox.information(
                self,
                "Exito",
                ("Se pudo abrir el archivo " + ubicacion)
            )
            self.ui.plainTextEdit.clear()
            self.ui.plainTextEdit.insertPlainText(str(self.__lista))
        else:
            QMessageBox.critical(
                self,
                "Error",
                ("No pudo abrir el archivo " + ubicacion)
            )

    @ Slot()
    def click_agregar_inicio(self):
        self.__lista.agregar_inicio(self.procesarParticula())
        self.__contador += 1

    @ Slot()
    def click_agregar_final(self):
        self.__lista.agregar_final(self.procesarParticula())
        self.__contador += 1

    @ Slot()
    def mostrar(self):
        self.ui.plainTextEdit.clear()
        self.ui.plainTextEdit.insertPlainText(str(self.__lista))

    def procesarParticula(self):
        """ id, origen_x, origen_y, destino_x, destino_y, veloicidad, red, green, blue,
distancia """
        return Particula(self.__contador,
                         self.ui.spnnOrigenX.value(),
                         self.ui.spnnOrigenY.value(),
                         self.ui.spnnDestinoX.value(),
                         self.ui.spnnDestinoY.value(),
                         self.ui.spnnVelocidad.text(),
                         self.ui.spnnRed.value(),
                         self.ui.spnnBlue.value(),
                         self.ui.spnnGreen.value(),
                         self.ui.spnnDistancia.value())
```

particular.py

```
from algoritmos import distancia_euclidiana


class Particula(object):
    __id = 0
    __origen_x = 0
    __origen_y = 0
    __destino_x = 0
    __destino_y = 0
    __veloicidad = 0
    __red = 0
    __green = 0
    __blue = 0
    __distancia = 0.0

    def __init__(self, id, origen_x, origen_y, destino_x, destino_y, veloicidad, red,
green, blue, distancia):
        """ Propiedades de la clase """
        self.__id = id
        self.__origen_x = origen_x
        self.__origen_y = origen_y
        self.__destino_x = destino_x
        self.__destino_y = destino_y
        self.__veloicidad = veloicidad
        self.__red = red
        self.__green = green
        self.__blue = blue
        self.__distancia = distancia
        """ Calculo de la distancia euclidiana """
        self.__distancia = distancia_euclidiana(
            origen_x, origen_y, destino_x, destino_y)
    """ Metodos geters """

    @property
    def id(self):
        return self.__id

    @property
    def origen_x(self):
        return self.__origen_x

    @property
    def origen_y(self):
        return self.__origen_y
```

```python
    @property
    def destino_x(self):
        return self.__destino_x

    @property
    def destino_y(self):
        return self.__destino_y

    @property
    def veloicidad(self):
        return self.__veloicidad

    @property
    def red(self):
        return self.__red

    @property
    def green(self):
        return self.__green

    @property
    def blue(self):
        return self.__blue

    @property
    def distancia(self):
        return self.__distancia

    def __str__(self):
        return (
            "#####################################\n"
            + "Id: " + str(self.__id) + ",\n"
            + "Origen X: " + str(self.__origen_x) + ",\n"
            + "Origen Y: " + str(self.__origen_y) + ",\n"
            + "Destino X: " + str(self.__destino_x) + ",\n"
            + "Destino Y: " + str(self.__destino_y) + ",\n"
            + "Velocidad: " + str(self.__veloicidad) + ",\n"
            + "Rojo: " + str(self.__red) + ",\n"
            + "Verde: " + str(self.__green) + ",\n"
            + "Azul: " + str(self.__blue) + ",\n"
            + "Distancia: " + str(self.__distancia))

    def to_dict(self):
        return {
```

```
            "id": self.__id,
            "origen_x": self.__origen_x,
            "origen_y": self.__origen_y,
            "destino_x": self.__destino_x,
            "destino_y": self.__destino_y,
            "veloicidad": self.__veloicidad,
            "red": self.__red,
            "green": self.__green,
            "blue": self.__blue,
            "distancia": + self.__distancia
        }
```

listaParticula.py

```
import json
from Particula import Particula


class listaParticula:
    def __init__(self):
        self.__particulas = []

    def agregar_inicio(self, particula: Particula):
        self.__particulas.insert(0, particula)

    def agregar_final(self, particula: Particula):
        self.__particulas.append(particula)

    def mostrar(self):
        for particula in self.__particulas:
            print(particula)

    def __str__(self):
        return "".join(
            str(particula) for particula in self.__particulas
        )

    def guardar(self, ubicacion):
        try:
            with open(ubicacion, 'w') as archivo:
                lista = [particula.to_dict()
                         for particula in self.__particulas]

                json.dump(lista, archivo, indent=5)
            return 1
        except:
```

```
            return 0

    def abrir(self, ubicacion):
        try:
            with open(ubicacion, 'r') as archivo:
                lista = json.load(archivo)
                self.__particulas = [Particula(**particula)
                                     for particula in lista]
            return 1
        except:
            return 0

    def __len__(self):
        return len(self.__particulas)

    def __iter__(self):
        self.cont = 0
        return self

    def __next__(self):
        if self.cont < len(self.__particulas):
            """ Asignamos la particula a devolver """
            particula = self.__particulas[self.cont]
            """ Incremenamos el contador """
            self.cont += 1
            return particula
        """ detemos la iteración si se sobrepasa el tamaño de la lista """
        raise StopIteration
```

main.py

```
from PySide2.QtWidgets import QApplication
from mainwindow import MainWindow
import sys
app = QApplication()
window = MainWindow()
window.show()
sys.exit(app.exec_())
```

Subida a gitHub

```
Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8
$ git init
Initialized empty Git repository in C:/Users/Armando/Documents/GitHub/Actividad 8/.git/

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git add .

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        src/

nothing added to commit but untracked files present (use "git add" to track)

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git add .

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   src/Main.py
        new file:   src/Particula.py
        new file:   src/__pycache__/Particula.cpython-310.pyc
        new file:   src/__pycache__/algoritmos.cpython-310.pyc
        new file:   src/__pycache__/listaParticulas.cpython-310.pyc
        new file:   src/__pycache__/mainwindow.cpython-310.pyc
        new file:   src/__pycache__/ui_mainwindow.cpython-310.pyc
        new file:   src/algoritmos.py
        new file:   src/listaParticulas.py
        new file:   src/mainwindow.py
        new file:   src/mainwindow.ui
        new file:   src/ui_mainwindow.py


Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git commit -m "Subida ilnicial"
[master (root-commit) 6985313] Subida ilnicial
 12 files changed, 648 insertions(+)
 create mode 100644 src/Main.py
 create mode 100644 src/Particula.py
 create mode 100644 src/__pycache__/Particula.cpython-310.pyc
 create mode 100644 src/__pycache__/algoritmos.cpython-310.pyc
 create mode 100644 src/__pycache__/listaParticulas.cpython-310.pyc
 create mode 100644 src/__pycache__/mainwindow.cpython-310.pyc
 create mode 100644 src/__pycache__/ui_mainwindow.cpython-310.pyc
 create mode 100644 src/algoritmos.py
 create mode 100644 src/listaParticulas.py
 create mode 100644 src/mainwindow.py
 create mode 100644 src/mainwindow.ui
 create mode 100644 src/ui_mainwindow.py
```

```
Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git config --global user.email diego.hernandez7503@alumnos.udg.mx

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git config --global user.name Diego-Armando-H

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git commit -m "Subida ilnicial"
On branch master
nothing to commit, working tree clean

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (master)
$ git branch -M main

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (main)
$ git remote add origin https://github.com/Diego-Armando-H/Actividad_8.git

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (main)
$ git push -u origin main
remote: Repository not found.
fatal: repository 'https://github.com/Diego-Armando-H/Actividad_8.git/' not found

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (main)
$ git remote remove origin

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (main)
$ git remote add origin https://github.com/Diego-Armando-H/Actividad-8.git

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (main)
$ git push -u origin main
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 4 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (16/16), 10.44 KiB | 1.74 MiB/s, done.
Total 16 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Diego-Armando-H/Actividad-8.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 8 (main)
$
```