# ›Fuerza Bruta‹

Cruz Collazo Wendy Paola.

*Seminario de solución de Problemas de Algoritmia.*

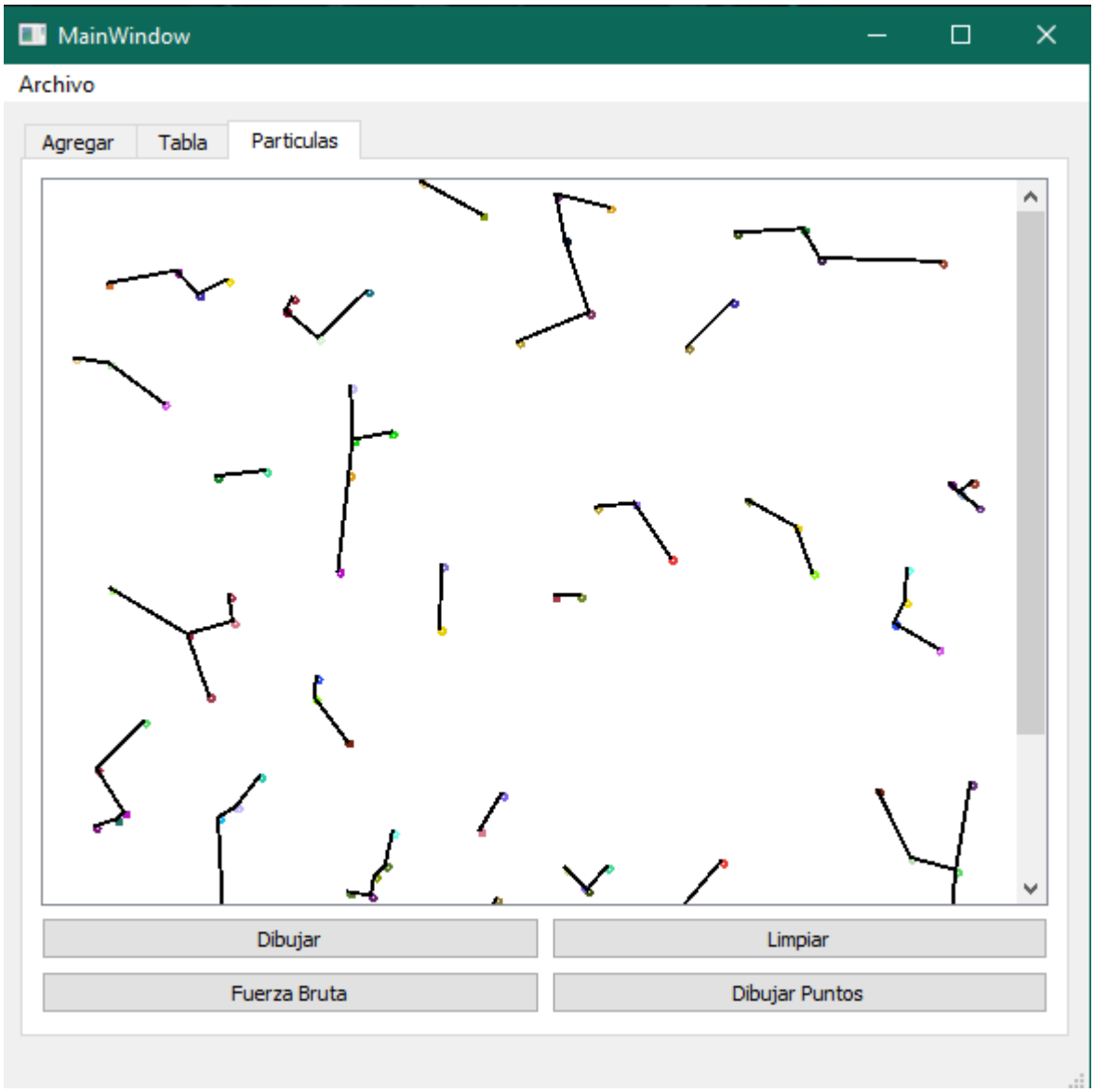
## Lineamientos de Evaluación:

- 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 los puntos de las partículas en el QScene.
- Se muestra captura de pantalla del resultado del algoritmo de fuerza bruta en el QScene.

## ✓ Desarrollo:

Función que dibuja los puntos de las partículas:

Función que realiza la Fuerza Bruta:




✓ Conclusiones:

Fue una actividad un poquito complicada al principio, pero después de ver algunos videos de referencia pude entenderle mejor y se me hizo un poco más fácil.

- ✓ Referencias:
  - Clase Fuerza Bruta – Michel Davalos Boites https://www.youtube.com/watch?v=zmPOdDMTk0Y
  - Clase Fuerza Bruta – Michel Davalos Boites https://www.youtube.com/watch?v=NUsjpfKaD_U

- ✓ Codigo:

## Administradora.py

```
from particula import Particula
from algoritmos import puntosCercanos
import json

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

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

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

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

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

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


    def __iter__(self):
        self.cont = 0

        return self
```

```
    def __next__(self):
        if self.cont < len(self.__particulas):
            particula = self.__particulas[self.cont]
            self.cont += 1
            return particula
        else:
            raise StopIteration

    def guardar(self,ubiacion):
        try:
            with open(ubiacion,'w') as archivo:
                lista = [particula.to_dict() for particula in
self.__particulas]
                json.dump(lista,archivo, indent = 5)
            return
        except:
            return 0
             #json.dump()

    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 ordenar_id(self):
        return self.__particulas.sort(key=lambda particula: particula.id)

    def ordenar_distancia(self):
        return self.__particulas.sort(key=lambda particula:
particula.distancia)

    def puntos(self):
        par = []
        par1= []
        for particula in self.__particulas:
            x1 = particula.origen_x
            y1 = particula.origen_y
            x2 = particula.destino_x
            y2 = particula.destino_y
```

```
            x = (x1, y1)
            y = (x2, y2)

            par1.append(y)
            par.append(x)

        lista = par1 + par

        return puntosCercanos(lista)
```

## Algoritmos.py

```
import math

def distancia_euclidiana(x_1, y_1, x_2, y_2):
    a = (x_2 - x_1)*(x_2 - x_1)
    b = (y_2 - y_1)*(y_2 - y_1)

    c = a + b

    distancia = math.sqrt(c)

    return distancia

def puntosCercanos(puntos:list)->list:
    resultado=[]
    for puntoi in puntos:
        x1 = puntoi[0]
        y1 = puntoi[1]
        min = 1000
        cercano = (0,0)
        for puntoj in puntos:
            if puntoi != puntoj:
                x2 = puntoj[0]
                y2= puntoj[1]
                dis = distancia_euclidiana(x1, y1, x2, y2)
                if dis < min:
                    min = dis
                    cercano = (x2, y2)
        resultado.append((puntoi, cercano))
    return resultado
```

## main.py

```
from PySide2.QtWidgets import QApplication
from mainwindow import MainWindow
import sys

app = QApplication()

window = MainWindow()

window.show()

sys.exit(app.exec_())
```

## Mainwindow.py

```
from re import S
from PySide2.QtWidgets import
QMainWindow,QFileDialog,QMessageBox,QTableWidgetItem, QGraphicsScene
from ui_mainwindow import Ui_MainWindow
from administradora import Administradora
from particula import Particula
from PySide2.QtCore import Slot
from PySide2.QtGui import QPen, QColor, QTransform

class MainWindow(QMainWindow):
    def __init__(self):
        super(MainWindow,self).__init__()

        self.administrador = Administradora()

        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.ui.Agregar_final.clicked.connect(self.agregar_final)
        self.ui.Agregar_Inicio.clicked.connect(self.agregar_inicio)
        self.ui.Mostrar.clicked.connect(self.ver)

        self.ui.actionAbrir.triggered.connect(self.action_abrir_archivo)
        self.ui.actionGuardar.triggered.connect(self.action_guardar_archivo)

        self.ui.view_button.clicked.connect(self.mostrar_tabla)
        self.ui.search_button.clicked.connect(self.buscar_tabla)

        self.ui.dibujar.clicked.connect(self.dibujar)
        self.ui.limpiar.clicked.connect(self.limipiar)
```

```
        self.scene = QGraphicsScene()
        self.ui.graphicsView.setScene(self.scene)

        self.ui.ordenar_Id.clicked.connect(self.Ordenar_Id)
        self.ui.ordenar_id2.clicked.connect(self.Ordenar_Id2)
        self.ui.ordenar_distancia.clicked.connect(self.Ordenar_Distancia)
        self.ui.ordenar_distancia2.clicked.connect(self.Ordenar_Distancia2)

        self.ui.fuerzaBruta.clicked.connect(self.FuerzaBruta)
        self.ui.dibujarPuntos.clicked.connect(self.DibujarPuntos)

    @Slot()
    def FuerzaBruta(self):
        pen = QPen()
        pen.setWidth(2)
        resultado = self.administrador.puntos()

        for punto1,punto2 in resultado:
            x1 = punto1[0]
            y1 = punto1[1]
            x2 = punto2[0]
            y2 = punto2[1]

            self.scene.addLine(x1,y1,x2,y2,pen)

    @Slot()
    def DibujarPuntos(self):
        self.scene.clear()
        pen = QPen()
        pen.setWidth(2)
        for particula in self.administrador:
            origenx = int(particula.origen_x)
            origeny = int(particula.origen_y)
            destinox = int(particula.destino_x)
            destinoy = int(particula.destino_y)
            red = int(particula.red)
            green = int(particula.green)
            blue= int(particula.blue)

            color = QColor(red,green,blue)
            pen.setColor(color)

            self.scene.addEllipse(origenx,origeny,3,3,pen)
            self.scene.addEllipse(destinox,destinoy,3,3,pen)
```

```
    @Slot ()
    def Ordenar_Id(self):
        self.ui.Print.clear()
        self.administrador.ordenar_id()
        self.ui.Print.insertPlainText(str(self.administrador))

    @Slot ()
    def Ordenar_Distancia(self):
        self.ui.Print.clear()
        self.administrador.ordenar_distancia()
        self.ui.Print.insertPlainText(str(self.administrador))

    @Slot ()
    def Ordenar_Id2(self):
        self.ui.table.clear()
        self.administrador.ordenar_id()
        self.ui.table.setColumnCount(9)
        headers = ["ID","Origen X","Origen Y","Destino X","Destino
Y","Red","Green","Blue","Distancia"]
        self.ui.table.setHorizontalHeaderLabels(headers)

        self.ui.table.setRowCount(len(self.administrador))

        row = 0
        for particula in self.administrador:
            id_widget = QTableWidgetItem(str(particula.id))
            origenx_widget = QTableWidgetItem(str(particula.origen_x))
            origeny_widget = QTableWidgetItem(str(particula.origen_y))
            destinox_widget = QTableWidgetItem(str(particula.destino_x))
            destinoy_widget = QTableWidgetItem(str(particula.destino_y))
            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.table.setItem(row,0,id_widget)
            self.ui.table.setItem(row,1,origenx_widget)
            self.ui.table.setItem(row,2,origeny_widget)
            self.ui.table.setItem(row,3,destinox_widget)
            self.ui.table.setItem(row,4,destinoy_widget)
            self.ui.table.setItem(row,5,red_widget)
            self.ui.table.setItem(row,6,green_widget)
            self.ui.table.setItem(row,7,blue_widget)
            self.ui.table.setItem(row,8,distancia_widget)
```

```
            row += 1

    @Slot ()
    def Ordenar_Distancia2(self):
        self.ui.table.clear()
        self.administrador.ordenar_distancia()
        self.ui.table.setColumnCount(9)
        headers = ["ID","Origen X","Origen Y","Destino X","Destino
Y","Red","Green","Blue","Distancia"]
        self.ui.table.setHorizontalHeaderLabels(headers)

        self.ui.table.setRowCount(len(self.administrador))

        row = 0
        for particula in self.administrador:
            id_widget = QTableWidgetItem(str(particula.id))
            origenx_widget = QTableWidgetItem(str(particula.origen_x))
            origeny_widget = QTableWidgetItem(str(particula.origen_y))
            destinox_widget = QTableWidgetItem(str(particula.destino_x))
            destinoy_widget = QTableWidgetItem(str(particula.destino_y))
            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.table.setItem(row,0,id_widget)
            self.ui.table.setItem(row,1,origenx_widget)
            self.ui.table.setItem(row,2,origeny_widget)
            self.ui.table.setItem(row,3,destinox_widget)
            self.ui.table.setItem(row,4,destinoy_widget)
            self.ui.table.setItem(row,5,red_widget)
            self.ui.table.setItem(row,6,green_widget)
            self.ui.table.setItem(row,7,blue_widget)
            self.ui.table.setItem(row,8,distancia_widget)

            row += 1

    @Slot()
    def wheelEvent(self, event):
        if event.delta() > 0:
            self.ui.graphicsView.scale(1.2, 1.2)
        else:
            self.ui.graphicsView.scale(0.8, 0.8)
```

```
    @Slot ()
    def dibujar(self):
        pen = QPen()
        pen.setWidth(3)

        for particula in self.administrador:
            origenx = int(particula.origen_x)
            origeny = int(particula.origen_y)
            destinox = int(particula.destino_x)
            destinoy = int(particula.destino_y)

            red = int(particula.red)
            green = int(particula.green)
            blue = int(particula.blue)

            color = QColor(red, green, blue)
            pen.setColor(color)

            self.scene.addEllipse(origenx, origeny, 3, 3, pen)
            self.scene.addEllipse(destinox, destinoy, 3, 3, pen)
            self.scene.addLine(origenx, origeny, destinox, destinoy, pen)

    @Slot()
    def limipiar(self):
        self.scene.clear()

    @Slot()
    def buscar_tabla(self):
        id = self.ui.search_line.text()
        encontrado = False

        for particula in self.administrador:

            if int(id)  ==  particula.id:
                self.ui.table.clear()
                self.ui.table.setRowCount(1)
                headers = ["ID","Origen X","Origen Y","Destino X","Destino
Y","Red","Green","Blue","Distancia"]
                self.ui.table.setHorizontalHeaderLabels(headers)

                id_widget = QTableWidgetItem(str(particula.id))
                origenx_widget = QTableWidgetItem(str(particula.origen_x))
                origeny_widget = QTableWidgetItem(str(particula.origen_y))
                destinox_widget = QTableWidgetItem(str(particula.destino_x))
                destinoy_widget = QTableWidgetItem(str(particula.destino_y))
```

```
                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.table.setItem(0,0,id_widget)
                self.ui.table.setItem(0,1,origenx_widget)
                self.ui.table.setItem(0,2,origeny_widget)
                self.ui.table.setItem(0,3,destinox_widget)
                self.ui.table.setItem(0,4,destinoy_widget)
                self.ui.table.setItem(0,5,red_widget)
                self.ui.table.setItem(0,6,green_widget)
                self.ui.table.setItem(0,7,blue_widget)
                self.ui.table.setItem(0,8,distancia_widget)

                encontrado = True

                return

        if not encontrado:
            QMessageBox.warning(self,'Atencíon',f'La particula con ID "{id}"
no fue encontrado')

    @Slot()
    def mostrar_tabla(self):
        self.ui.table.setColumnCount(9)
        headers = ["ID","Origen X","Origen Y","Destino X","Destino
Y","Red","Green","Blue","Distancia"]
        self.ui.table.setHorizontalHeaderLabels(headers)

        self.ui.table.setRowCount(len(self.administrador))

        row = 0
        for particula in self.administrador:
            id_widget = QTableWidgetItem(str(particula.id))
            origenx_widget = QTableWidgetItem(str(particula.origen_x))
            origeny_widget = QTableWidgetItem(str(particula.origen_y))
            destinox_widget = QTableWidgetItem(str(particula.destino_x))
            destinoy_widget = QTableWidgetItem(str(particula.destino_y))
            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.table.setItem(row,0,id_widget)
            self.ui.table.setItem(row,1,origenx_widget)
            self.ui.table.setItem(row,2,origeny_widget)
            self.ui.table.setItem(row,3,destinox_widget)
            self.ui.table.setItem(row,4,destinoy_widget)
            self.ui.table.setItem(row,5,red_widget)
            self.ui.table.setItem(row,6,green_widget)
            self.ui.table.setItem(row,7,blue_widget)
            self.ui.table.setItem(row,8,distancia_widget)

            row += 1

    @Slot()
    def action_abrir_archivo(self):
        ubicacion = QFileDialog.getOpenFileName(self,'Abrir
Archivo','.','JSON (*.json)')[0]
        if self.administrador.abrir(ubicacion):
            QMessageBox.information(self,"Exito","Se abrió el archivo de" +
ubicacion)
        else:
            QMessageBox.information(self,"Error","No se pudo abrir el
archivo de " + ubicacion)


    @Slot()
    def action_guardar_archivo(self):
        ubicacion = QFileDialog.getSaveFileName(self,'Guardar
Archivo','.','JSON (*.json)')[0]
        if self.administrador.guardar(ubicacion):
            QMessageBox.information(self,"Exito","Se creó el archivo con
exito en " + ubicacion)
        else:
            QMessageBox.information(self,"Error","No se pudo crear el
archivo en " + ubicacion)

    @Slot()
    def ver(self):
        self.ui.Print.clear()
        self.ui.Print.insertPlainText(str(self.administrador))

    @Slot()
    def agregar_final(self):
        ID = self.ui.ID_spinBox.value()
        OrigenX = self.ui.OrigenX_spinBox.value()
        OrigenY = self.ui.OrigenY_spinBox.value()
```

```
        DestinoX = self.ui.DestinoX_spinBox.value()
        DestinoY = self.ui.DestinoY_spinBox.value()
        Red = self.ui.Red_spinBox.value()
        Green = self.ui.Green_spinBox.value()
        Blue = self.ui.Blue_spinBox.value()

        particula1 =
Particula(ID,OrigenX,OrigenY,DestinoX,DestinoY,Red,Green,Blue)
        self.administrador.agregar_final(particula1)

    @Slot()
    def agregar_inicio(self):
        ID = self.ui.ID_spinBox.value()
        OrigenX = self.ui.OrigenX_spinBox.value()
        OrigenY = self.ui.OrigenY_spinBox.value()
        DestinoX = self.ui.DestinoX_spinBox.value()
        DestinoY = self.ui.DestinoY_spinBox.value()
        Red = self.ui.Red_spinBox.value()
        Green = self.ui.Green_spinBox.value()
        Blue = self.ui.Blue_spinBox.value()

        particula1 =
Particula(ID,OrigenX,OrigenY,DestinoX,DestinoY,Red,Green,Blue)
        self.administrador.agregar_inicio(particula1)
```

## particula.py

```
from algoritmos import distancia_euclidiana

class Particula:
    def __init__(self,id = 0, origen_x = 0, origen_y = 0, destino_x = 0,
destino_y=0,red = 0, green = 0, blue = 0):
        self.__id = id
        self.__origen_x = origen_x
        self.__origen_y = origen_y
        self.__destino_x = destino_x
        self.__destino_y = destino_y
        self.__red = red
        self.__green = green
        self.__blue = blue
        self.__distancia =
distancia_euclidiana(origen_x,origen_y,destino_x,destino_y)

    def __str__(self):
```

```
        return('Id : ' + str(self.__id) + '\n' + 'Origen en X :' +
str(self.__origen_x) + '\n' +
               'Origen en Y :' + str(self.__origen_y) + '\n' + 'Destino en X
:' + str(self.__destino_x) + '\n' +
               'Destino en Y: ' + str(self.__destino_y) + '\n' + 'Distancia
: ' + str(self.__distancia) + '\n' +
               'Red :' + str(self.__red) + '\n' 'Green :' +
str(self.__green) + '\n' 'Blue :' + str(self.__blue) + '\n')

    @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

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

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

    @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 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,
            "red": self.__red,
            "green": self.__green,
            "blue": self.__blue
        }
```

## Ui_mainwindow.py

```
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(583, 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_2 = QGridLayout(self.tab)
        self.gridLayout_2.setObjectName(u"gridLayout_2")
        self.groupBox = QGroupBox(self.tab)
        self.groupBox.setObjectName(u"groupBox")
        self.gridLayout = QGridLayout(self.groupBox)
        self.gridLayout.setObjectName(u"gridLayout")
        self.label_2 = QLabel(self.groupBox)
        self.label_2.setObjectName(u"label_2")
```

```
        self.gridLayout.addWidget(self.label_2, 6, 0, 1, 1)

        self.label0 = QLabel(self.groupBox)
        self.label0.setObjectName(u"label0")

        self.gridLayout.addWidget(self.label0, 1, 0, 1, 1)

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

        self.gridLayout.addWidget(self.label_5, 5, 0, 1, 1)

        self.OrigenY_spinBox = QSpinBox(self.groupBox)
        self.OrigenY_spinBox.setObjectName(u"OrigenY_spinBox")
        self.OrigenY_spinBox.setMaximum(999)

        self.gridLayout.addWidget(self.OrigenY_spinBox, 3, 1, 1, 1)

        self.Blue_spinBox = QSpinBox(self.groupBox)
        self.Blue_spinBox.setObjectName(u"Blue_spinBox")

        self.gridLayout.addWidget(self.Blue_spinBox, 8, 1, 1, 1)

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

        self.gridLayout.addWidget(self.label_3, 3, 0, 1, 1)

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

        self.gridLayout.addWidget(self.label_4, 4, 0, 1, 1)

        self.Mostrar = QPushButton(self.groupBox)
        self.Mostrar.setObjectName(u"Mostrar")

        self.gridLayout.addWidget(self.Mostrar, 11, 0, 1, 2)

        self.Red_spinBox = QSpinBox(self.groupBox)
        self.Red_spinBox.setObjectName(u"Red_spinBox")

        self.gridLayout.addWidget(self.Red_spinBox, 6, 1, 1, 1)

        self.DestinoX_spinBox = QSpinBox(self.groupBox)
        self.DestinoX_spinBox.setObjectName(u"DestinoX_spinBox")
```

```
        self.DestinoX_spinBox.setMaximum(255)

        self.gridLayout.addWidget(self.DestinoX_spinBox, 4, 1, 1, 1)

        self.Agregar_Inicio = QPushButton(self.groupBox)
        self.Agregar_Inicio.setObjectName(u"Agregar_Inicio")

        self.gridLayout.addWidget(self.Agregar_Inicio, 9, 0, 1, 2)

        self.ordenar_Id = QPushButton(self.groupBox)
        self.ordenar_Id.setObjectName(u"ordenar_Id")

        self.gridLayout.addWidget(self.ordenar_Id, 12, 0, 1, 2)

        self.ID_spinBox = QSpinBox(self.groupBox)
        self.ID_spinBox.setObjectName(u"ID_spinBox")
        self.ID_spinBox.setMaximum(999)

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

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

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

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

        self.gridLayout.addWidget(self.label_8, 8, 0, 1, 1)

        self.Agregar_final = QPushButton(self.groupBox)
        self.Agregar_final.setObjectName(u"Agregar_final")

        self.gridLayout.addWidget(self.Agregar_final, 10, 0, 1, 2)

        self.Green_spinBox = QSpinBox(self.groupBox)
        self.Green_spinBox.setObjectName(u"Green_spinBox")

        self.gridLayout.addWidget(self.Green_spinBox, 7, 1, 1, 1)

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

        self.gridLayout.addWidget(self.label_7, 7, 0, 1, 1)
```

```
        self.OrigenX_spinBox = QSpinBox(self.groupBox)
        self.OrigenX_spinBox.setObjectName(u"OrigenX_spinBox")
        self.OrigenX_spinBox.setMaximum(999)

        self.gridLayout.addWidget(self.OrigenX_spinBox, 1, 1, 1, 1)

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

        self.gridLayout.addWidget(self.DestinoY_spinBox, 5, 1, 1, 1)

        self.ordenar_distancia = QPushButton(self.groupBox)
        self.ordenar_distancia.setObjectName(u"ordenar_distancia")

        self.gridLayout.addWidget(self.ordenar_distancia, 13, 0, 1, 2)


        self.gridLayout_2.addWidget(self.groupBox, 0, 0, 1, 1)

        self.Print = QPlainTextEdit(self.tab)
        self.Print.setObjectName(u"Print")

        self.gridLayout_2.addWidget(self.Print, 0, 1, 1, 1)

        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QWidget()
        self.tab_2.setObjectName(u"tab_2")
        self.gridLayout_4 = QGridLayout(self.tab_2)
        self.gridLayout_4.setObjectName(u"gridLayout_4")
        self.table = QTableWidget(self.tab_2)
        self.table.setObjectName(u"table")

        self.gridLayout_4.addWidget(self.table, 0, 0, 1, 4)

        self.horizontalSpacer = QSpacerItem(40, 20, QSizePolicy.Expanding,
QSizePolicy.Minimum)

        self.gridLayout_4.addItem(self.horizontalSpacer, 2, 0, 1, 1)

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

        self.gridLayout_4.addWidget(self.view_button, 1, 3, 1, 1)
```

```
        self.search_button = QPushButton(self.tab_2)
        self.search_button.setObjectName(u"search_button")

        self.gridLayout_4.addWidget(self.search_button, 1, 2, 1, 1)

        self.search_line = QLineEdit(self.tab_2)
        self.search_line.setObjectName(u"search_line")

        self.gridLayout_4.addWidget(self.search_line, 1, 0, 1, 2)

        self.ordenar_distancia2 = QPushButton(self.tab_2)
        self.ordenar_distancia2.setObjectName(u"ordenar_distancia2")

        self.gridLayout_4.addWidget(self.ordenar_distancia2, 2, 3, 1, 1)

        self.ordenar_id2 = QPushButton(self.tab_2)
        self.ordenar_id2.setObjectName(u"ordenar_id2")

        self.gridLayout_4.addWidget(self.ordenar_id2, 2, 2, 1, 1)

        self.tabWidget.addTab(self.tab_2, "")
        self.tab_3 = QWidget()
        self.tab_3.setObjectName(u"tab_3")
        self.gridLayout_5 = QGridLayout(self.tab_3)
        self.gridLayout_5.setObjectName(u"gridLayout_5")
        self.limpiar = QPushButton(self.tab_3)
        self.limpiar.setObjectName(u"limpiar")

        self.gridLayout_5.addWidget(self.limpiar, 1, 1, 1, 1)

        self.dibujar = QPushButton(self.tab_3)
        self.dibujar.setObjectName(u"dibujar")

        self.gridLayout_5.addWidget(self.dibujar, 1, 0, 1, 1)

        self.graphicsView = QGraphicsView(self.tab_3)
        self.graphicsView.setObjectName(u"graphicsView")

        self.gridLayout_5.addWidget(self.graphicsView, 0, 0, 1, 2)

        self.fuerzaBruta = QPushButton(self.tab_3)
        self.fuerzaBruta.setObjectName(u"fuerzaBruta")

        self.gridLayout_5.addWidget(self.fuerzaBruta, 2, 0, 1, 1)
```

```
        self.dibujarPuntos = QPushButton(self.tab_3)
        self.dibujarPuntos.setObjectName(u"dibujarPuntos")

        self.gridLayout_5.addWidget(self.dibujarPuntos, 2, 1, 1, 1)

        self.tabWidget.addTab(self.tab_3, "")

        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, 583, 21))
        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(2)


        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("MainWindo
w", u"Ctrl+D", None))
#endif // QT_CONFIG(shortcut)
        self.groupBox.setTitle(QCoreApplication.translate("MainWindow",
u"Particulas", None))
        self.label_2.setText(QCoreApplication.translate("MainWindow",
u"Red", None))
        self.label0.setText(QCoreApplication.translate("MainWindow",
u"Origen X", None))
        self.label_5.setText(QCoreApplication.translate("MainWindow",
u"Destino Y", None))
        self.label_3.setText(QCoreApplication.translate("MainWindow",
u"Origen Y", None))
        self.label_4.setText(QCoreApplication.translate("MainWindow",
u"Destino X", None))
        self.Mostrar.setText(QCoreApplication.translate("MainWindow",
u"Mostrar", None))
        self.Agregar_Inicio.setText(QCoreApplication.translate("MainWindow",
u"Agregar al inicio", None))
        self.ordenar_Id.setText(QCoreApplication.translate("MainWindow",
u"Ordenar Id", None))
        self.label.setText(QCoreApplication.translate("MainWindow", u"ID:",
None))
        self.label_8.setText(QCoreApplication.translate("MainWindow",
u"Blue", None))
        self.Agregar_final.setText(QCoreApplication.translate("MainWindow",
u"Agregar al final", None))
        self.label_7.setText(QCoreApplication.translate("MainWindow",
u"Green", None))
        self.ordenar_distancia.setText(QCoreApplication.translate("MainWindo
w", u"Ordenar Distancia", None))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab),
QCoreApplication.translate("MainWindow", u"Agregar", None))
        self.view_button.setText(QCoreApplication.translate("MainWindow",
u"Mostrar", None))
        self.search_button.setText(QCoreApplication.translate("MainWindow",
u"Buscar", None))
        self.search_line.setPlaceholderText(QCoreApplication.translate("Main
Window", u"ID de la particula", None))
        self.ordenar_distancia2.setText(QCoreApplication.translate("MainWind
ow", u"Ordenar Distancia", None))
        self.ordenar_id2.setText(QCoreApplication.translate("MainWindow",
u"Ordenar Id", None))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2),
QCoreApplication.translate("MainWindow", u"Tabla", None))
```

```
        self.limpiar.setText(QCoreApplication.translate("MainWindow",
u"Limpiar", None))
        self.dibujar.setText(QCoreApplication.translate("MainWindow",
u"Dibujar", None))
        self.fuerzaBruta.setText(QCoreApplication.translate("MainWindow",
u"Fuerza Bruta", None))
        self.dibujarPuntos.setText(QCoreApplication.translate("MainWindow",
u"Dibujar Puntos", None))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3),
QCoreApplication.translate("MainWindow", u"Particulas", None))
        self.menuArchivo.setTitle(QCoreApplication.translate("MainWindow",
u"Archivo", None))
    # retranslateUi
```