Fuerza Bruta

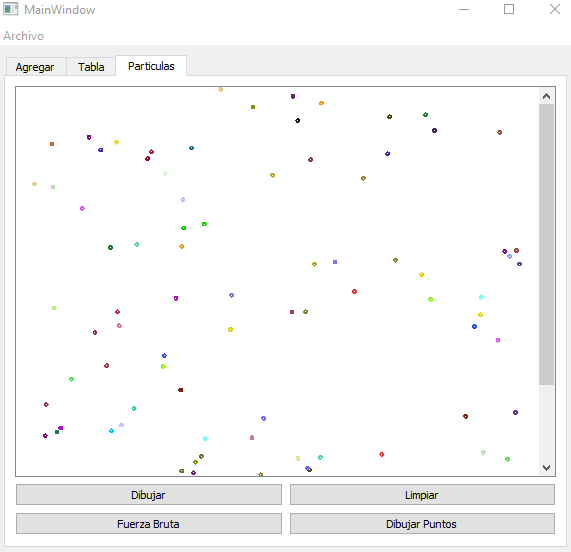
Cruz Collazo Wendy Paola.

*Seminario de solución de Problemas de Algoritmia.*

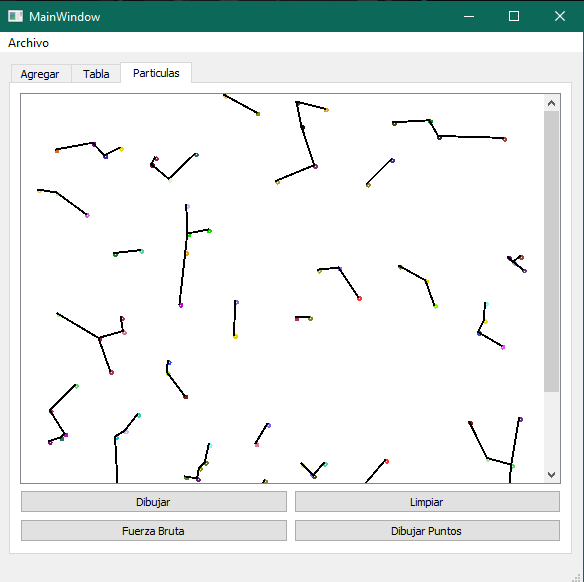
**Lineamientos de Evaluación:**

* 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("MainWindow", 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("MainWindow", 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("MainWindow", u"ID de la particula", None))

        self.ordenar\_distancia2.setText(QCoreApplication.translate("MainWindow", 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