Actividad 07

(QFileDialog)

*Alcaraz Valdivia Marcos Fernando*

*Seminario de Solución de Problemas de Algoritmia*

**Lineamientos de evaluación**

* El reporte está en formato Google Docs o PDF. *(REALIZADO)*
* El reporte sigue las pautas del Formato de Actividades. *(REALIZADO)*
* El reporte tiene desarrollada todas las pautas del Formato de Actividades. *(REALIZADO)*
* Se muestra la captura de pantalla de las partículas con el método mostrar () previo a generar el respaldo. *(REALIZADO)*
* Se muestran capturas de pantallas de los pasos que se realizan en la interfaz para generar el respaldo. *(REALIZADO)*
* Se muestra el contenido del archivo “.json”. *(REALIZADO)*
* Se muestran capturas de pantallas de los pasos que se realizan en la interfaz para abrir el archivo de respaldo “.json”. *(REALIZADO)*
* Se muestra la captura de pantalla de las partículas con el método mostrar () después de abrir el respaldo. *(REALIZADO)*

***AAAAAAAAAA  
Satisfacción 10/10***

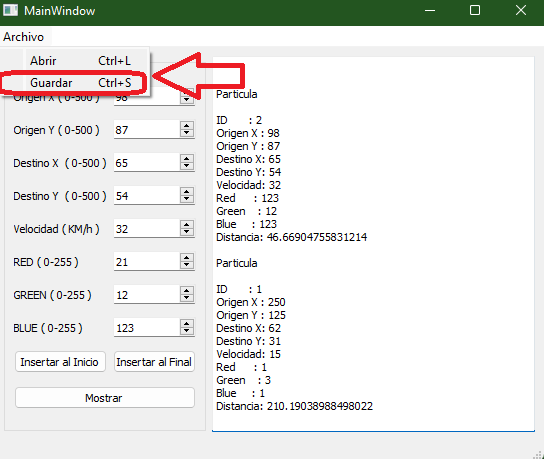
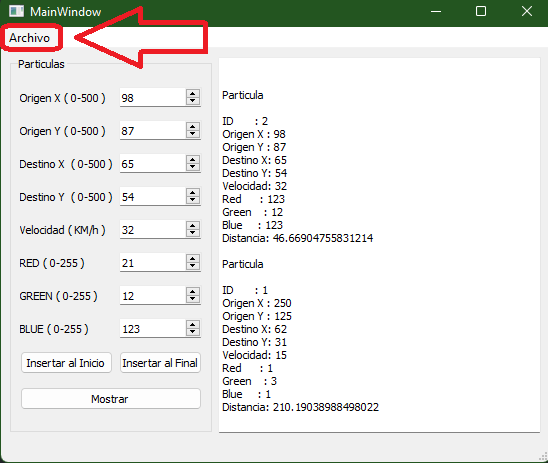
***Desarrollo***

***Captura de pantalla previo a generar respaldo***

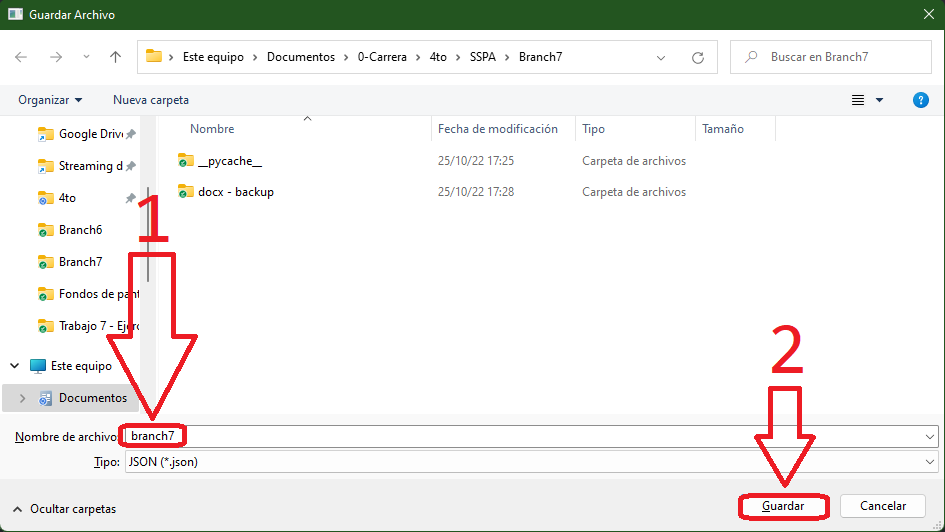
Interfaz de usuario gráfica

Descripción generada automáticamente

***Pasos para realizar en la interfaz para realizar un respaldo***

******

***Se selecciona una ruta y nombre del archivo***

******

***Recibe el mensaje de respuesta al guardar el archivo***

Interfaz de usuario gráfica

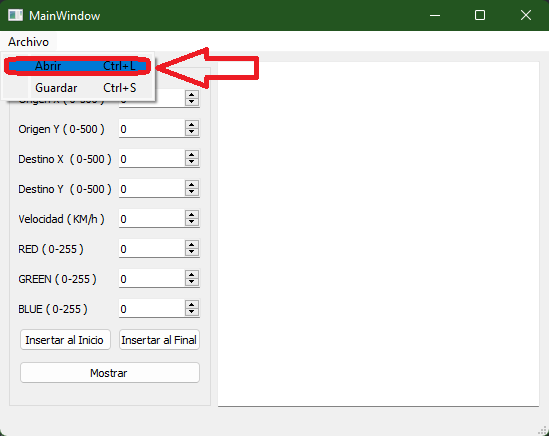
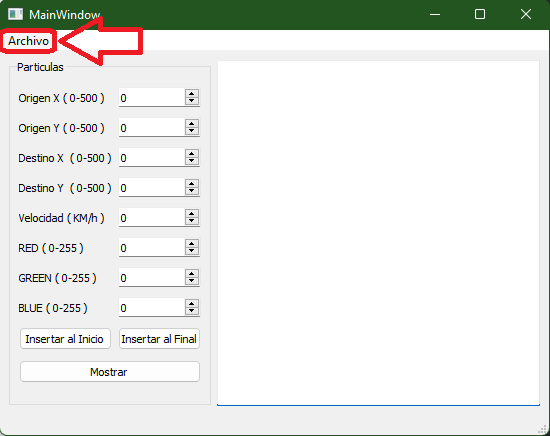
Descripción generada automáticamente con confianza media

***Contenido del archivo JSON***

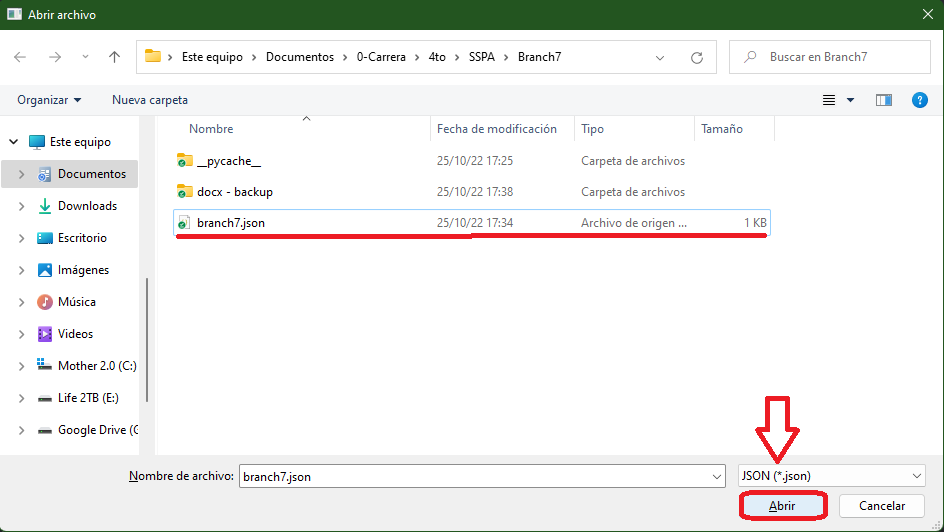
Texto

Descripción generada automáticamente

***Pasos para abrir archivo***

******

***Seleccionar archivo JSON***

******

***Recibir mensaje al abrir el archivo***

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

***Captura de pantalla después de realizar el respaldo***

Interfaz de usuario gráfica

Descripción generada automáticamente

***Conclusiones***

Usar Python es divertido.

Todo lo que hacía en C++ era gran numero de veces más complicado que en Python. Dios, leer archivos JSON con solo una librería que Python ya tiene por defecto, leer y escribir archivos de manera tan simple, ***recorrer listas completas con solo un ciclo for***, es increíble.

Estoy consiente de que he sido capaz de realizar todo este proyecto hasta ahora gracias a Michel Davalos Boites, pero debo decir que es realmente más fácil resolver un problema en Python que en C++.

***Referencias***

# **PySide2 - QFileDialog (Qt for Python)(IV)**

<https://www.youtube.com/watch?v=HRY8QvXmcDM>

***Código***

***Se mostrarán solo los archivos modificados***

***particula.py***

# Archivo 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, velocidad=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.\_\_velocidad = velocidad

        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 ('\n\nParticula\n' +

                '\nID       : ' + str(self.\_\_id) +

                '\nOrigen X : ' + str(self.\_\_origen\_x) +

                '\nOrigen Y : ' + str(self.\_\_origen\_y) +

                '\nDestino X: ' + str(self.\_\_destino\_x) +

                '\nDestino Y: ' + str(self.\_\_destino\_y) +

                '\nVelocidad: ' + str(self.\_\_velocidad) +

                '\nRed      : ' + str(self.\_\_blue) +

                '\nGreen    : ' + str(self.\_\_green) +

                '\nBlue     : ' + str(self.\_\_blue) +

                '\nDistancia: ' + 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,

            "velocidad": self.\_\_velocidad,

            "red": self.\_\_red,

            "green": self.\_\_green,

            "blue": self.\_\_blue,

        }

***manager.py***

# Archivo manager.py

from particula import Particula

import json

class Manager:

    def \_\_init\_\_(self):

        self.\_\_particulas = []

    def agregarInicio(self, particula: Particula):

        self.\_\_particulas.insert(0, particula)

    def agregarFinal(self, particula: Particula):

        self.\_\_particulas.append(particula)

    def imprimir(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

***mainwindow.py***

from PySide2.QtWidgets import QMainWindow, QFileDialog, QMessageBox

from PySide2.QtCore import Slot

from ui\_mainwindow import Ui\_MainWindow

from manager import Manager

from particula import Particula

class MainWindow(QMainWindow):

    def \_\_init\_\_(self):

        super(MainWindow, self).\_\_init\_\_()

        self.manager = Manager()

        self.ui = Ui\_MainWindow()

        self.ui.setupUi(self)

        self.id = 0

        # Cuando el botón pushbutton es presionado, ejecuta la función click\_agregar

        # self.ui.mostrar.clicked.connect(self.click\_mostrar)

        self.ui.insertar\_inicio.clicked.connect(self.click\_insertar\_inicio)

        self.ui.insertar\_final.clicked.connect(self.click\_insertar\_final)

        self.ui.mostrar.clicked.connect(self.click\_mostrar)

        self.ui.accionAbrir.triggered.connect(self.accionAbrirArchivo)

        self.ui.accionGuardar.triggered.connect(self.accionGuardarArchivo)

    @Slot()

    def accionAbrirArchivo(self):

        ubicacion = QFileDialog.getOpenFileName(

            self,

            "Abrir archivo",

            ".",

            "JSON (\*.json)"

        )[0]

        if self.manager.abrir(ubicacion):

            self.click\_mostrar()

            QMessageBox.information(self, "Abrir archivo", "Archivo abierto Exitosamente : " + ubicacion)

        else:

            QMessageBox.critical(self, "Error", "No se puede abrir el archivo : " + ubicacion)

    @Slot()

    def accionGuardarArchivo(self):

        ubicacion = QFileDialog.getSaveFileName(

            self,

            "Guardar Archivo",

            ".",

            "JSON (\*.json)"

        )[0]

        if self.manager.guardar(ubicacion):

            QMessageBox.information(self, "Archivo Guardado", "Guardado Exitoso : " + ubicacion)

        else:

            QMessageBox.critical(self, "Error", "Archivo no Guardado : " + ubicacion)

    @Slot()

    def click\_insertar\_inicio(self):

        self.id += 1

        aux = Particula(self.id, self.ui.ox.value(), self.ui.oy.value(), self.ui.dx.value(), self.ui.dy.value(

        ), self.ui.velocidad.value(), self.ui.red.value(), self.ui.green.value(), self.ui.blue.value())

        self.manager.agregarInicio(aux)

        self.click\_mostrar()

    @Slot()

    def click\_insertar\_final(self):

        self.id += 1

        aux = Particula(self.id, self.ui.ox.value(), self.ui.oy.value(), self.ui.dx.value(), self.ui.dy.value(

        ), self.ui.velocidad.value(), self.ui.red.value(), self.ui.green.value(), self.ui.blue.value())

        self.manager.agregarFinal(aux)

        self.click\_mostrar()

    @Slot()

    def click\_mostrar(self):

        self.ui.lista\_particulas.clear()

        self.ui.lista\_particulas.insertPlainText(str(self.manager))

***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(549, 405)

        self.accionGuardar = QAction(MainWindow)

        self.accionGuardar.setObjectName(u"accionGuardar")

        self.accionAbrir = QAction(MainWindow)

        self.accionAbrir.setObjectName(u"accionAbrir")

        self.centralwidget = QWidget(MainWindow)

        self.centralwidget.setObjectName(u"centralwidget")

        self.gridLayout\_2 = QGridLayout(self.centralwidget)

        self.gridLayout\_2.setObjectName(u"gridLayout\_2")

        self.lista\_particulas = QPlainTextEdit(self.centralwidget)

        self.lista\_particulas.setObjectName(u"lista\_particulas")

        self.gridLayout\_2.addWidget(self.lista\_particulas, 0, 1, 1, 1)

        self.groupBox = QGroupBox(self.centralwidget)

        self.groupBox.setObjectName(u"groupBox")

        self.gridLayout = QGridLayout(self.groupBox)

        self.gridLayout.setObjectName(u"gridLayout")

        self.green = QSpinBox(self.groupBox)

        self.green.setObjectName(u"green")

        self.green.setMaximum(255)

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

        self.dy = QSpinBox(self.groupBox)

        self.dy.setObjectName(u"dy")

        self.dy.setMaximum(500)

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

        self.label\_7 = QLabel(self.groupBox)

        self.label\_7.setObjectName(u"label\_7")

        self.gridLayout.addWidget(self.label\_7, 1, 0, 1, 1)

        self.blue = QSpinBox(self.groupBox)

        self.blue.setObjectName(u"blue")

        self.blue.setMaximum(255)

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

        self.label\_5 = QLabel(self.groupBox)

        self.label\_5.setObjectName(u"label\_5")

        self.gridLayout.addWidget(self.label\_5, 6, 0, 1, 1)

        self.insertar\_final = QPushButton(self.groupBox)

        self.insertar\_final.setObjectName(u"insertar\_final")

        self.gridLayout.addWidget(self.insertar\_final, 8, 1, 1, 1)

        self.label\_3 = QLabel(self.groupBox)

        self.label\_3.setObjectName(u"label\_3")

        self.gridLayout.addWidget(self.label\_3, 4, 0, 1, 1)

        self.label\_2 = QLabel(self.groupBox)

        self.label\_2.setObjectName(u"label\_2")

        self.gridLayout.addWidget(self.label\_2, 3, 0, 1, 1)

        self.red = QSpinBox(self.groupBox)

        self.red.setObjectName(u"red")

        self.red.setMaximum(255)

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

        self.mostrar = QPushButton(self.groupBox)

        self.mostrar.setObjectName(u"mostrar")

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

        self.label\_4 = QLabel(self.groupBox)

        self.label\_4.setObjectName(u"label\_4")

        self.gridLayout.addWidget(self.label\_4, 5, 0, 1, 1)

        self.label\_6 = QLabel(self.groupBox)

        self.label\_6.setObjectName(u"label\_6")

        self.gridLayout.addWidget(self.label\_6, 7, 0, 1, 1)

        self.velocidad = QSpinBox(self.groupBox)

        self.velocidad.setObjectName(u"velocidad")

        self.velocidad.setMaximum(999)

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

        self.label = QLabel(self.groupBox)

        self.label.setObjectName(u"label")

        self.gridLayout.addWidget(self.label, 2, 0, 1, 1)

        self.dx = QSpinBox(self.groupBox)

        self.dx.setObjectName(u"dx")

        self.dx.setMaximum(500)

        self.gridLayout.addWidget(self.dx, 2, 1, 1, 1)

        self.insertar\_inicio = QPushButton(self.groupBox)

        self.insertar\_inicio.setObjectName(u"insertar\_inicio")

        self.gridLayout.addWidget(self.insertar\_inicio, 8, 0, 1, 1)

        self.label\_8 = QLabel(self.groupBox)

        self.label\_8.setObjectName(u"label\_8")

        self.gridLayout.addWidget(self.label\_8, 0, 0, 1, 1)

        self.oy = QSpinBox(self.groupBox)

        self.oy.setObjectName(u"oy")

        self.oy.setMaximum(500)

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

        self.ox = QSpinBox(self.groupBox)

        self.ox.setObjectName(u"ox")

        self.ox.setMaximum(500)

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

        self.gridLayout\_2.addWidget(self.groupBox, 0, 0, 1, 1)

        MainWindow.setCentralWidget(self.centralwidget)

        self.menubar = QMenuBar(MainWindow)

        self.menubar.setObjectName(u"menubar")

        self.menubar.setGeometry(QRect(0, 0, 549, 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.accionAbrir)

        self.menuArchivo.addAction(self.accionGuardar)

        self.retranslateUi(MainWindow)

        QMetaObject.connectSlotsByName(MainWindow)

    # setupUi

    def retranslateUi(self, MainWindow):

        MainWindow.setWindowTitle(QCoreApplication.translate("MainWindow", u"MainWindow", None))

        self.accionGuardar.setText(QCoreApplication.translate("MainWindow", u"Guardar", None))

#if QT\_CONFIG(shortcut)

        self.accionGuardar.setShortcut(QCoreApplication.translate("MainWindow", u"Ctrl+S", None))

#endif // QT\_CONFIG(shortcut)

        self.accionAbrir.setText(QCoreApplication.translate("MainWindow", u"Abrir", None))

#if QT\_CONFIG(shortcut)

        self.accionAbrir.setShortcut(QCoreApplication.translate("MainWindow", u"Ctrl+L", None))

#endif // QT\_CONFIG(shortcut)

        self.groupBox.setTitle(QCoreApplication.translate("MainWindow", u"Particulas", None))

        self.label\_7.setText(QCoreApplication.translate("MainWindow", u"Origen Y ( 0-500 )", None))

        self.label\_5.setText(QCoreApplication.translate("MainWindow", u"GREEN ( 0-255 )", None))

        self.insertar\_final.setText(QCoreApplication.translate("MainWindow", u"Insertar al Final", None))

        self.label\_3.setText(QCoreApplication.translate("MainWindow", u"Velocidad ( KM/h )", None))

        self.label\_2.setText(QCoreApplication.translate("MainWindow", u"Destino Y  ( 0-500 )", None))

        self.mostrar.setText(QCoreApplication.translate("MainWindow", u"Mostrar", None))

        self.label\_4.setText(QCoreApplication.translate("MainWindow", u"RED ( 0-255 )", None))

        self.label\_6.setText(QCoreApplication.translate("MainWindow", u"BLUE ( 0-255 )", None))

        self.label.setText(QCoreApplication.translate("MainWindow", u"Destino X  ( 0-500 )", None))

        self.insertar\_inicio.setText(QCoreApplication.translate("MainWindow", u"Insertar al Inicio", None))

        self.label\_8.setText(QCoreApplication.translate("MainWindow", u"Origen X ( 0-500 )", None))

        self.menuArchivo.setTitle(QCoreApplication.translate("MainWindow", u"Archivo", None))

    # retranslateUi