Міністерство освіти і науки України

**Прикарпатський національний університет**

**імені В.Стефаника**

*Факультет математики та інформатики*

*Кафедра інформаційних технологій*

*Людинно-машинна взаємодія*

Лабораторна робота № 8

Тема: Робота з контейнерами в середовищі Qt Creator: QList, QLinkedList

*Варіант 2*

Виконав: ***Гук Д.П.***

Група ІПЗ-31

Дата:12 листопада 2023 р.

Викладач: Пікуляк М.В.

Івано-Франківськ – 2023

**Мета роботи:**

Отримати навички роботи з файлами в ICP “Qt-Creator”.

**Завдання для виконання :**

**Завдання 1.** Дано файл f, компоненти якого є цілими числами. Запишіть в

файл g найбільше значення перших п’яти компонентів файлу f, потім – таких

п’яти компонентів і т.д. Якщо в останній групі виявиться менше п’яти

компонентів, то останній компонент файлу g має дорівнювати найбільшому з

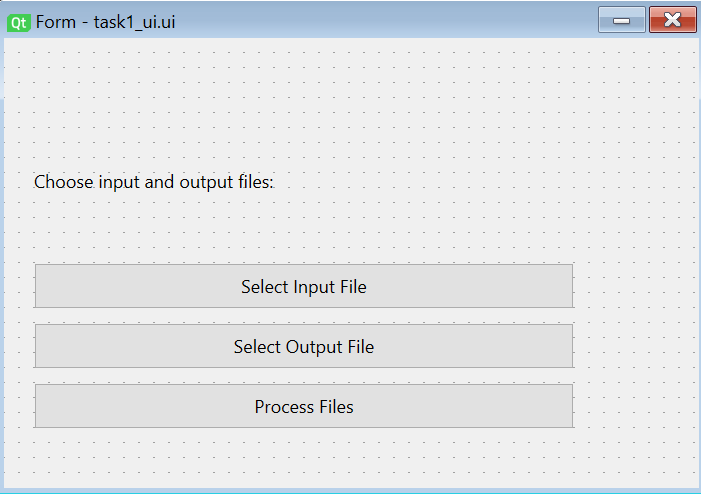
компонентів файлу f, що утворюють останню (неповну) групу.

**Завдання 2.** Дано символьні файли f1 і f2. Перепишіть зі збереженням

порядку проходження компоненти файлу f1 в файл f2, а компоненти файлу f2 – в файл f1. Використовуйте допоміжний файл h.

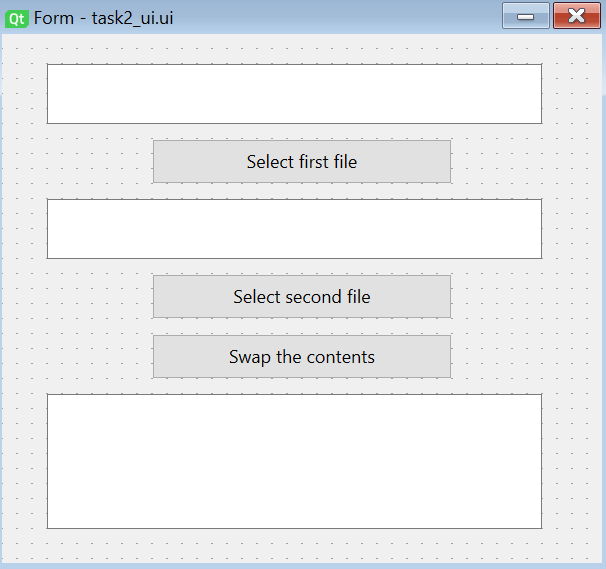
**Тексти скриптів і зображення діалогових вікон QtCreator з виконаними завданнями :**

**Завдання №1:**

****

import sys  
from PyQt6 import QtCore, QtGui, QtWidgets  
from PyQt6.QtWidgets import QFileDialog  
  
class FileProcessor:  
 def \_\_init\_\_(self, ui):  
 self.input\_filename = None  
 self.output\_filename = None  
  
 self.ui = ui  
 self.setup\_connections()  
  
 def setup\_connections(self):  
 self.ui.pushButton.clicked.connect(self.get\_input\_file)  
 self.ui.pushButton\_2.clicked.connect(self.get\_output\_file)  
 self.ui.pushButton\_3.clicked.connect(self.process\_files)  
  
 def get\_input\_file(self):  
 self.input\_filename, \_ = QFileDialog.getOpenFileName(None, 'Select Input File')  
 if self.input\_filename:  
 self.ui.label.setText(f'Input File: {self.input\_filename}')  
  
 def get\_output\_file(self):  
 self.output\_filename, \_ = QFileDialog.getSaveFileName(None, 'Select Output File')  
 if self.output\_filename:  
 self.ui.label.setText(f'Output File: {self.output\_filename}')  
  
 def process\_files(self):  
 if not self.input\_filename or not self.output\_filename:  
 return  
  
 try:  
 with open(self.input\_filename, 'r') as file:  
 numbers = [int(line) for line in file.readlines()]  
 chunk\_size = 5  
 with open(self.output\_filename, 'w') as output\_file:  
 while numbers:  
 chunk = sorted(numbers[:chunk\_size], reverse=True)  
 for num in chunk:  
 output\_file.write(str(num) + '\n')  
 numbers = numbers[chunk\_size:]  
 except FileNotFoundError:  
 self.ui.label.setText('File not found. Please select valid files.')  
 return  
  
 self.ui.label.setText('Files processed successfully.')  
  
class Ui\_Form(object):  
 def setupUi(self, Form):  
 Form.setObjectName("Form")  
 Form.resize(400, 300)  
 self.label = QtWidgets.QLabel(parent=Form)  
 self.label.setGeometry(QtCore.QRect(20, 70, 171, 31))  
 self.label.setObjectName("label")  
 self.pushButton = QtWidgets.QPushButton(parent=Form)  
 self.pushButton.setGeometry(QtCore.QRect(20, 150, 360, 31))  
 self.pushButton.setObjectName("pushButton")  
 self.pushButton\_2 = QtWidgets.QPushButton(parent=Form)  
 self.pushButton\_2.setGeometry(QtCore.QRect(20, 190, 360, 31))  
 self.pushButton\_2.setObjectName("pushButton\_2")  
 self.pushButton\_3 = QtWidgets.QPushButton(parent=Form)  
 self.pushButton\_3.setGeometry(QtCore.QRect(20, 230, 360, 31))  
 self.pushButton\_3.setObjectName("pushButton\_3")  
  
 self.retranslateUi(Form)  
 QtCore.QMetaObject.connectSlotsByName(Form)  
  
 def retranslateUi(self, Form):  
 \_translate = QtCore.QCoreApplication.translate  
 Form.setWindowTitle(\_translate("Form", "Task 1"))  
 self.label.setText(\_translate("Form", "Choose input and output files: "))  
 self.pushButton.setText(\_translate("Form", "Select Input File"))  
 self.pushButton\_2.setText(\_translate("Form", "Select Output File"))  
 self.pushButton\_3.setText(\_translate("Form", "Process Files"))  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QtWidgets.QApplication(sys.argv)  
 Form = QtWidgets.QWidget()  
 ui = Ui\_Form()  
 ui.setupUi(Form)  
 Form.show()  
  
 processor = FileProcessor(ui)  
 sys.exit(app.exec())

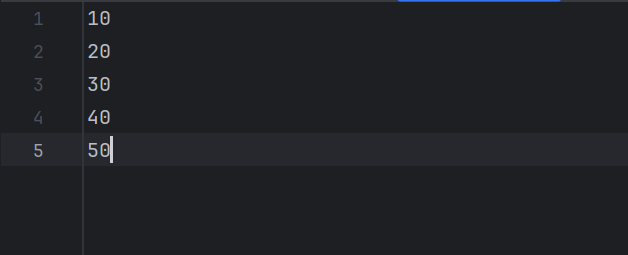
**Завдання №2:**

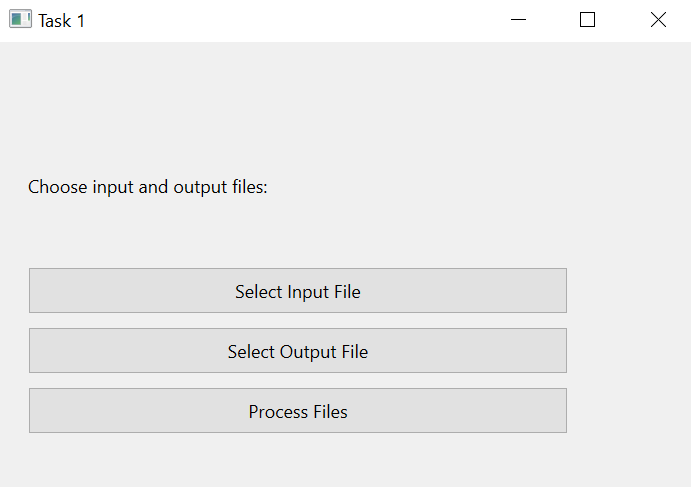


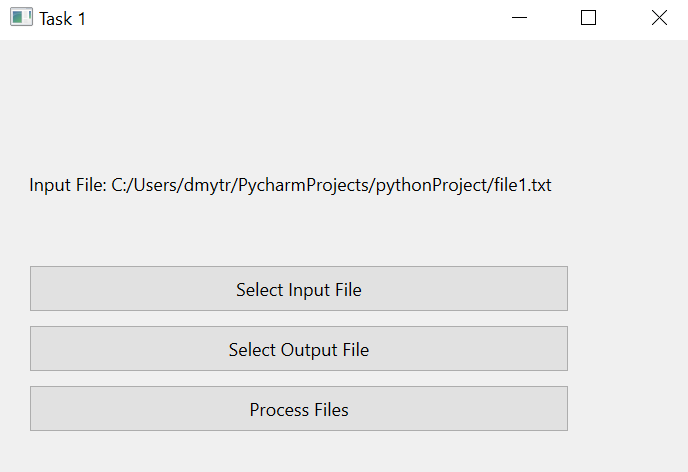
from PyQt6 import QtCore, QtGui, QtWidgets  
from PyQt6.QtWidgets import QFileDialog  
import shutil  
  
class FileSwapper:  
 def \_\_init\_\_(self, ui):  
 self.ui = ui  
 self.setup\_connections()  
  
 def setup\_connections(self):  
 self.ui.pushButton.clicked.connect(self.select\_first\_file)  
 self.ui.pushButton\_2.clicked.connect(self.select\_second\_file)  
 self.ui.pushButton\_3.clicked.connect(self.swap\_contents)  
  
 def select\_first\_file(self):  
 file\_path, \_ = QFileDialog.getOpenFileName(None, "Select first file", "", "Text Files (\*.txt);;All Files (\*)")  
 self.ui.lineEdit.setText(file\_path)  
  
 def select\_second\_file(self):  
 file\_path, \_ = QFileDialog.getOpenFileName(None, "Select second file", "", "Text Files (\*.txt);;All Files (\*)")  
 self.ui.lineEdit\_2.setText(file\_path)  
  
 def swap\_contents(self):  
 first\_file\_path = self.ui.lineEdit.text()  
 second\_file\_path = self.ui.lineEdit\_2.text()  
  
 if first\_file\_path == "" or second\_file\_path == "":  
 self.ui.lineEdit\_3.setText("Please select both files.")  
 return  
  
 try:  
 temp\_file\_path = "temp.txt"  
 shutil.copy(first\_file\_path, temp\_file\_path)  
 shutil.copy(second\_file\_path, first\_file\_path)  
 shutil.copy(temp\_file\_path, second\_file\_path)  
 self.ui.lineEdit\_3.setText("Files swapped successfully.")  
  
 except Exception as e:  
 self.ui.lineEdit\_3.setText(f"Error: {e}")  
  
class Ui\_Form(object):  
 def setupUi(self, Form):  
 Form.setObjectName("Form")  
 Form.resize(400, 353)  
 self.lineEdit = QtWidgets.QLineEdit(parent=Form)  
 self.lineEdit.setGeometry(QtCore.QRect(30, 20, 330, 40))  
 self.lineEdit.setObjectName("lineEdit")  
 self.pushButton = QtWidgets.QPushButton(parent=Form)  
 self.pushButton.setGeometry(QtCore.QRect(100, 70, 200, 30))  
 self.pushButton.setObjectName("pushButton")  
 self.pushButton\_2 = QtWidgets.QPushButton(parent=Form)  
 self.pushButton\_2.setGeometry(QtCore.QRect(100, 160, 200, 30))  
 self.pushButton\_2.setObjectName("pushButton\_2")  
 self.lineEdit\_2 = QtWidgets.QLineEdit(parent=Form)  
 self.lineEdit\_2.setGeometry(QtCore.QRect(30, 110, 330, 40))  
 self.lineEdit\_2.setObjectName("lineEdit\_2")  
 self.pushButton\_3 = QtWidgets.QPushButton(parent=Form)  
 self.pushButton\_3.setGeometry(QtCore.QRect(100, 200, 200, 30))  
 self.pushButton\_3.setObjectName("pushButton\_3")  
 self.lineEdit\_3 = QtWidgets.QLineEdit(parent=Form)  
 self.lineEdit\_3.setGeometry(QtCore.QRect(30, 240, 330, 90))  
 self.lineEdit\_3.setObjectName("lineEdit\_3")  
  
 self.retranslateUi(Form)  
 QtCore.QMetaObject.connectSlotsByName(Form)  
  
 def retranslateUi(self, Form):  
 \_translate = QtCore.QCoreApplication.translate  
 Form.setWindowTitle(\_translate("Form", "Task 2"))  
 self.pushButton.setText(\_translate("Form", "Select first file"))  
 self.pushButton\_2.setText(\_translate("Form", "Select second file"))  
 self.pushButton\_3.setText(\_translate("Form", "Swap the contents"))  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QtWidgets.QApplication([])  
 Form = QtWidgets.QWidget()  
 ui = Ui\_Form()  
 ui.setupUi(Form)  
 Form.show()  
  
 file\_swapper = FileSwapper(ui) # Initialize the FileSwapper passing the Ui\_Form instance  
 app.exec()

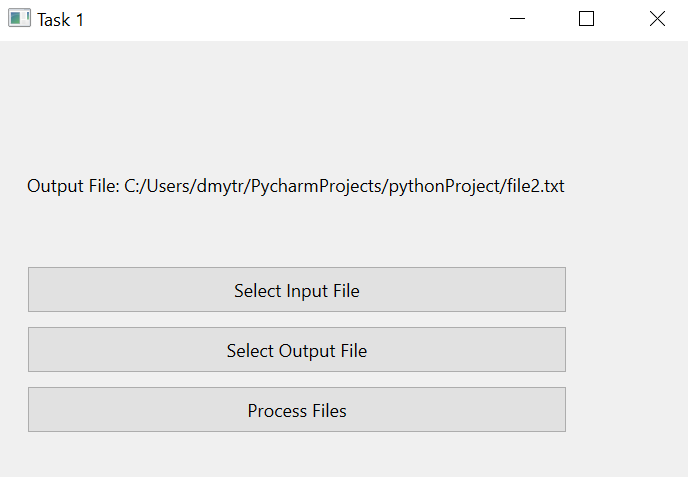
**Cкрін-шоти виконання завдань лабораторної роботи :**

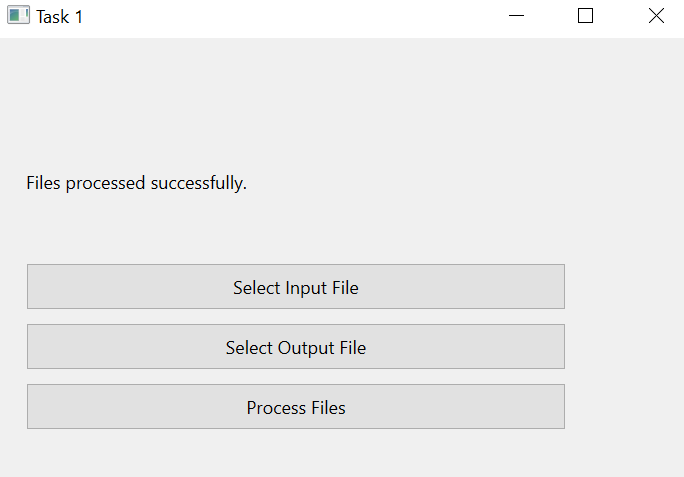
**Завдання №1 :**

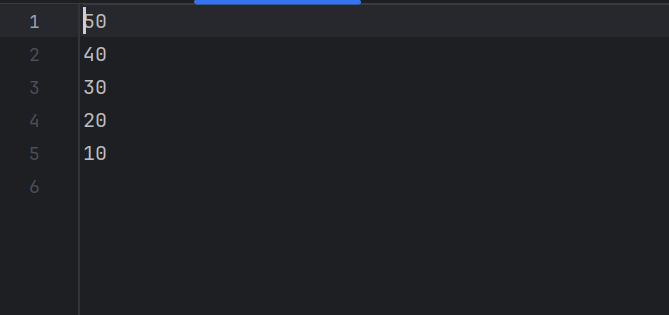












**Завдання №2:**

