Міністерство освіти і науки України
Прикарпатський національний університет
імені В.Стефаника

Факультет математики та інформатики
Кафедра інформаційних технологій

Людинно-машинна взаємодія

Лабораторна робота № 7

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.

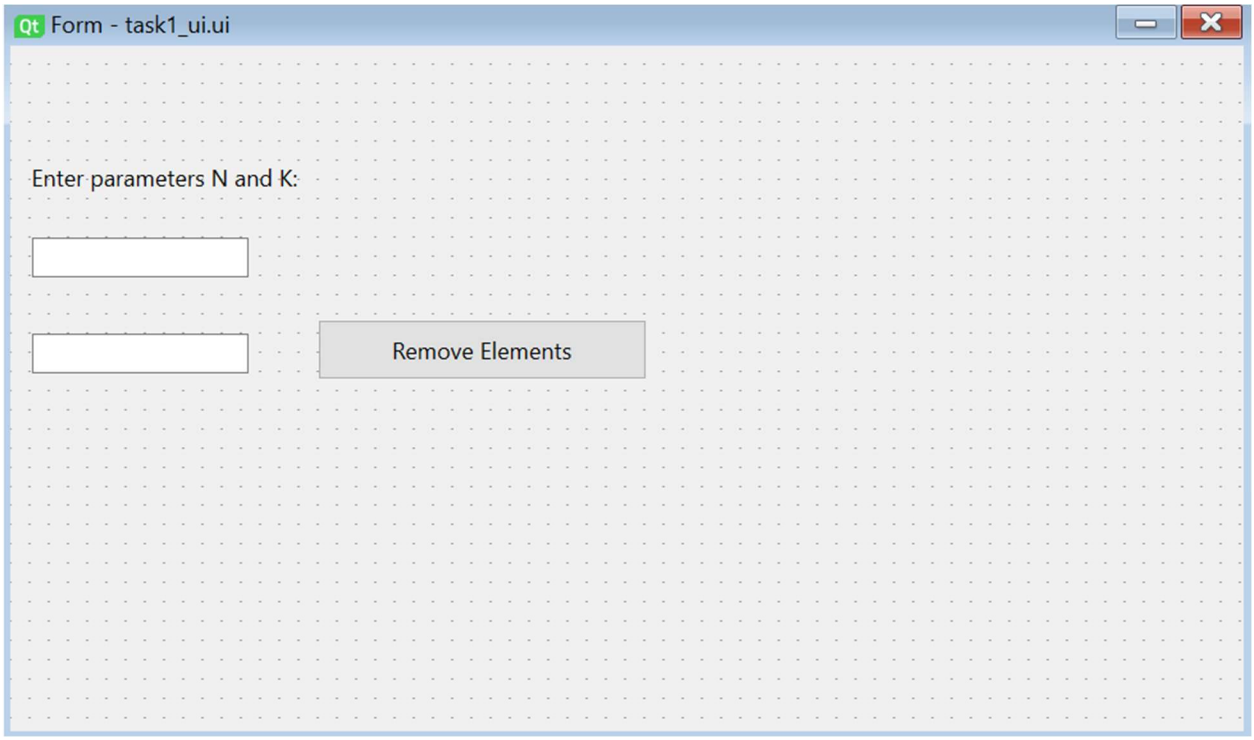
Івано-Франківськ – 2023

Мета роботи:
Отримати навички роботи з контейнерами QList, QLinkedList в ІСР “Qt-Creator”.

Завдання для виконання :
1. Заповнити список випадковими елементами і реалізувати видалення елементів з позицій з N по K.
2. Заповнити зв’язний список випадковими елементами і впорядкувати їх за спаданням.
3. Створити консольний проект для роботи з двохзв’язним списком (оголосіть екземпляр класу QLinkedList <int> list). Заповніть його випадковими значеннями і продемонструйте в ньому роботу таких алгоритмів STL:
а) count (first, last, value) – повертає значення, яке показує скільки разів елемент зі значенням value входить в послідовність, задану ітераторами;
б) reverse (first, last) – переставляє елементи в зворотному порядку;
в) iter_swap (first, last) – міняє місцями значення елементів, на які вказують ітератори.

Тексти скриптів і зображення діалогових вікон QtCreator з виконаними завданнями :

Завдання №1:

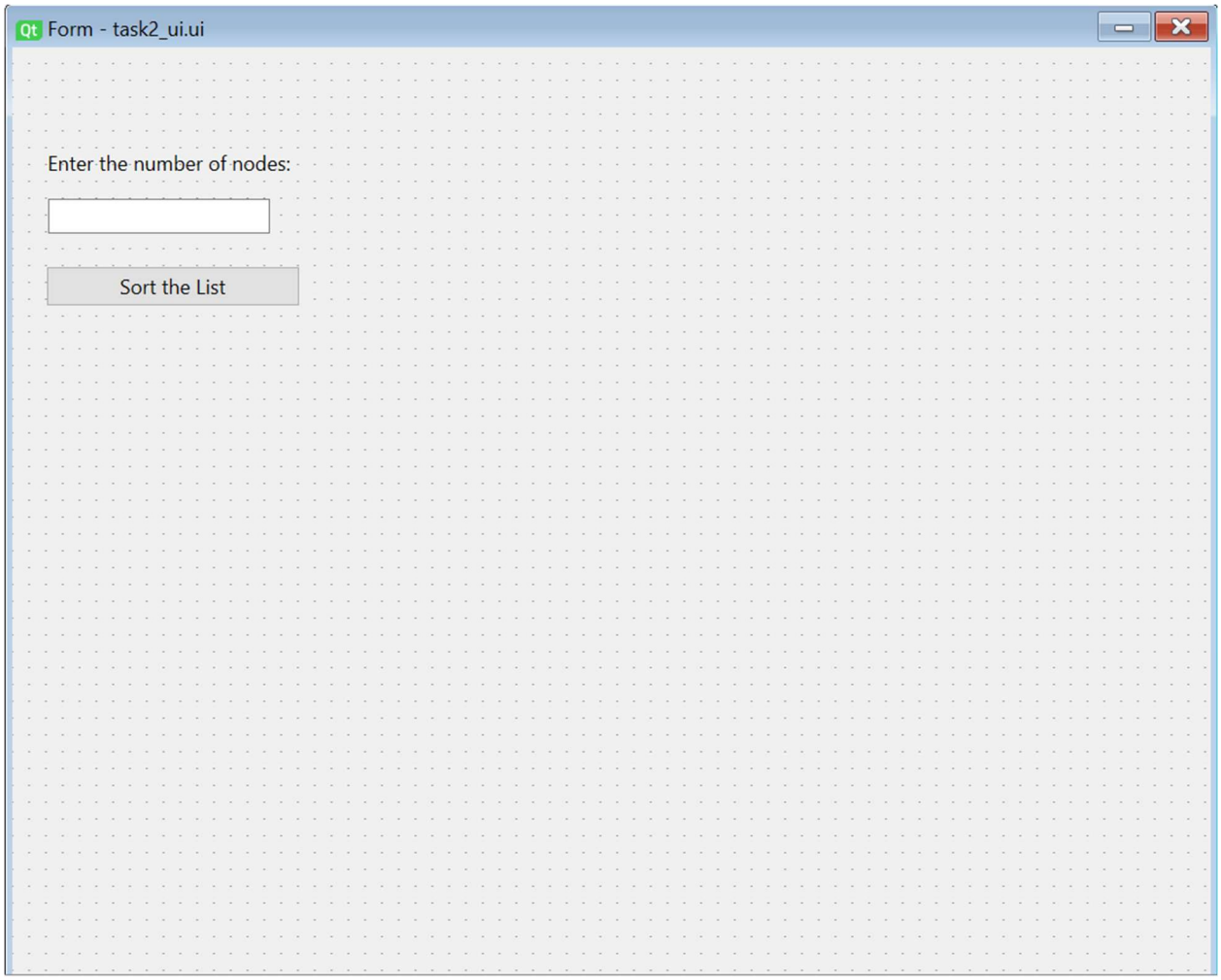



```
import random
import sys

from PyQt6 import QtCore, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(642, 357)
        self.label = QtWidgets.QLabel(parent=Form)
        self.label.setGeometry(QtCore.QRect(11, 61,
151, 16))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(parent=Form)
        self.label_2.setGeometry(QtCore.QRect(10, 270,
551, 61))
        self.label_2.setText("")
        self.label_2.setObjectName("label_2")
```

```
        self.lineEdit =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit.setGeometry(QtCore.QRect(11, 100,
113, 21))
        self.lineEdit.setObjectName("lineEdit")
        self.lineEdit_2 =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit_2.setGeometry(QtCore.QRect(11,
150, 113, 21))
        self.lineEdit_2.setObjectName("lineEdit_2")
        self.pushButton =
QtWidgets.QPushButton(parent=Form)
        self.pushButton.setGeometry(QtCore.QRect(160,
143, 171, 31))
        self.pushButton.setObjectName("pushButton")

self.pushButton.clicked.connect(self.remove_elements)
        self.label_3 = QtWidgets.QLabel(parent=Form)
        self.label_3.setGeometry(QtCore.QRect(10, 200,
551, 61))
        self.label_3.setText("")
        self.label_3.setObjectName("label_3")

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form",
"Task1"))
        self.label.setText(_translate("Form", "Enter
parameters N and K: "))
        self.pushButton.setText(_translate("Form",
"Remove Elements"))
```

```
    def remove_elements(self):
        try:
            N = int(self.lineEdit.text())
            K = int(self.lineEdit_2.text())

            if N >= K:
                self.label_2.setText("N must be less
than K")
                return

            random_list = [random.randint(1, 100) for _
in range(20)]
            self.label_3.setText("Random List: " + ',
'.join(map(str, random_list)))

            if K > len(random_list):
                K = len(random_list)

            del random_list[N - 1:K]

            self.label_2.setText("Updated List: " + ',
'.join(map(str, random_list)))

        except ValueError:
            self.label_2.setText("Please enter valid
integers for N and K")

app = QtWidgets.QApplication(sys.argv)
task3_dialog = QtWidgets.QDialog()
ui = Ui_Form()
ui.setupUi(task3_dialog)
task3_dialog.show()
sys.exit(app.exec())
```

Завдання №2:




```
import random
import sys
from collections.abc import Iterable

from PyQt6 import QtCore, QtWidgets

class Node:
    def __init__(self, data=None):
        self.data = data
        self.next = None

class LinkedList(Iterable):
```

```
    def __init__(self):
        self.head = None

    def __iter__(self):
        current = self.head
        while current:
            yield current.data
            current = current.next

    def append(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        current = self.head
        while current.next:
            current = current.next
        current.next = new_node

    def sort_descending(self):
        if not self.head or not self.head.next:
            return

        def merge_sort(node):
            if not node or not node.next:
                return node

            middle = get_middle(node)
            next_to_middle = middle.next
            middle.next = None

            left = merge_sort(node)
            right = merge_sort(next_to_middle)

            return merge(left, right)
```

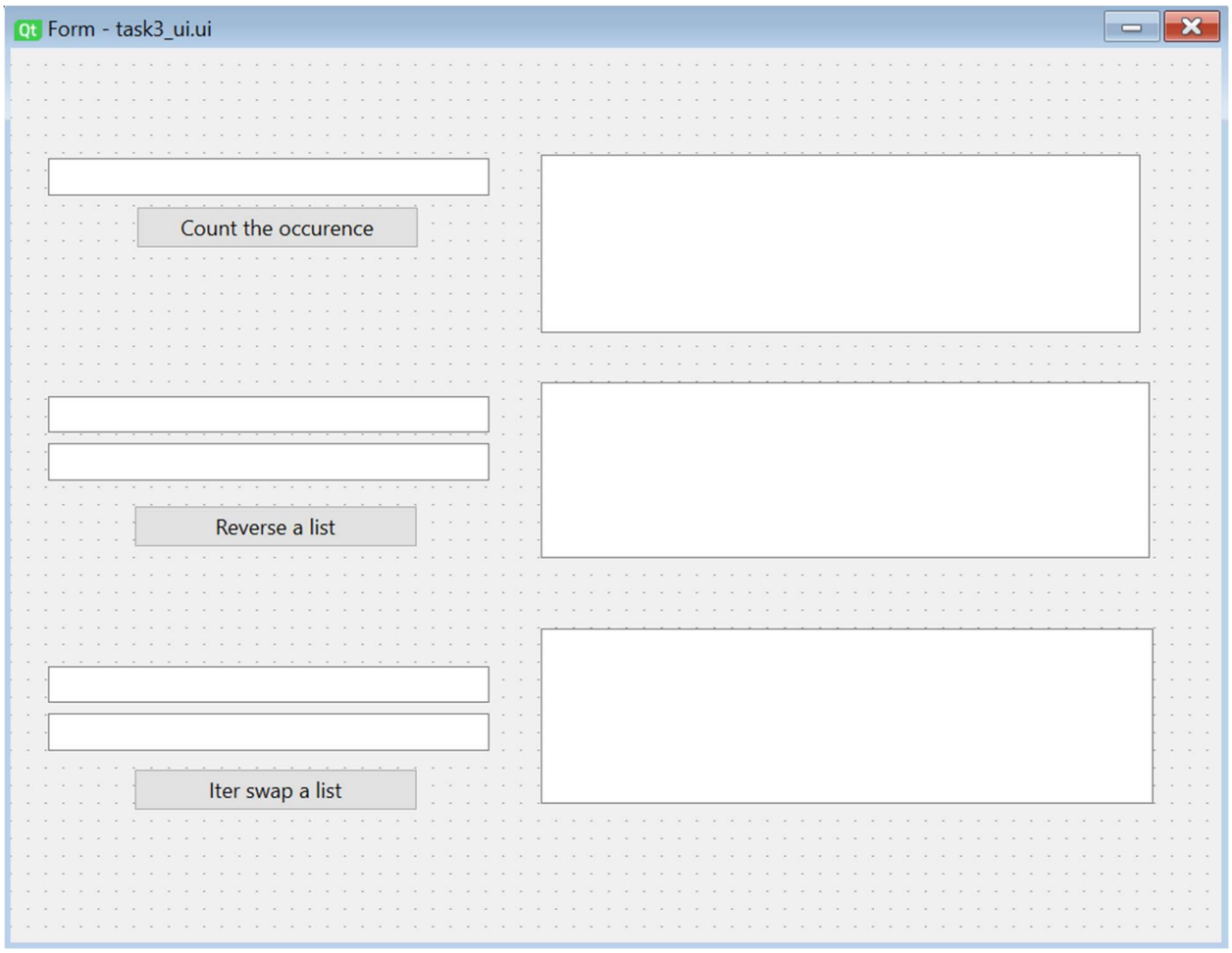
```
        def merge(left, right):
            result = None

            if not left:
                return right
            if not right:
                return left

            if left.data >= right.data:
                result = left
                result.next = merge(left.next, right)
            else:
                result = right
                result.next = merge(left, right.next)

            return result

        def get_middle(node):
            if not node:
                return node

            slow = node
            fast = node

            while fast.next and fast.next.next:
                slow = slow.next
                fast = fast.next.next

            return slow

        self.head = merge_sort(self.head)


class Ui_Form(object):
```

```
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(713, 571)
        self.label = QtWidgets.QLabel(parent=Form)
        self.label.setGeometry(QtCore.QRect(21, 61,
171, 16))
        self.label.setObjectName("label")
        self.lineEdit =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit.setGeometry(QtCore.QRect(21, 90,
132, 21))
        self.lineEdit.setObjectName("lineEdit")
        self.pushButton =
QtWidgets.QPushButton(parent=Form)
        self.pushButton.setGeometry(QtCore.QRect(20,
130, 151, 24))
        self.pushButton.setObjectName("pushButton")

self.pushButton.clicked.connect(self.sort_linked_list)
        self.label_2 = QtWidgets.QLabel(parent=Form)
        self.label_2.setGeometry(QtCore.QRect(21, 200,
301, 321))
        self.label_2.setText("")
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(parent=Form)
        self.label_3.setGeometry(QtCore.QRect(350, 200,
331, 321))
        self.label_3.setText("")
        self.label_3.setObjectName("label_3")

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
```

```
        Form.setWindowTitle(_translate("Form", "Task
2"))
        self.label.setText(_translate("Form", "Enter
the number of nodes: "))
        self.pushButton.setText(_translate("Form",
"Sort the List"))


    def sort_linked_list(self):
        try:
            num_nodes = int(self.lineEdit.text())

            linked_list = LinkedList()
            for _ in range(num_nodes):
                linked_list.append(random.randint(1,
100))

            original_list = "Original List:\n" + ',
'.join(
                str(node) + ('\n' if i % 10 == 9 else
'') for i, node in enumerate(linked_list))
            linked_list.sort_descending()
            sorted_list = "Sorted List (descending):\n"
+ ', '.join(
                str(node) + ('\n' if i % 10 == 9 else
'') for i, node in enumerate(linked_list))

            self.label_2.setText(original_list)
            self.label_3.setText(sorted_list)

        except ValueError:
            self.label_2.setText("Please enter a valid
integer for the number of nodes")
            self.label_3.setText("Please enter a valid
integer for the number of nodes")
```

```
app = QtWidgets.QApplication(sys.argv)
task2 = QtWidgets.QDialog()
ui = Ui_Form()
ui.setupUi(task2)
task2.show()
sys.exit(app.exec())
```

Завдання №3:




```
import random

from PyQt6 import QtCore, QtWidgets
```

```
class Node:
    def __init__(self, data):
        self.data = data
        self.prev = None
        self.next = None

class DoublyLinkedList:
    def __init__(self):
        self.head = None
        self.tail = None

    def append(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            self.tail = new_node
        else:
            new_node.prev = self.tail
            self.tail.next = new_node
            self.tail = new_node

    def count(self, value):
        count = 0
        current = self.head
        while current:
            if current.data == value:
                count += 1
            current = current.next
        return count

    def reverse(self, start_idx, end_idx,
callback=None):
        if start_idx < 0 or end_idx < 0 or start_idx >=
end_idx:
```

```
            return

        current = self.head
        count = 0

        while count < start_idx:
            current = current.next
            count += 1

        start_node = current

        while count < end_idx:
            current = current.next
            count += 1

        end_node = current

        while start_node != end_node and end_node.next
!= start_node:
            start_node.data, end_node.data =
end_node.data, start_node.data
            start_node = start_node.next
            end_node = end_node.prev

            if callback:
                callback()
                QtCore.QCoreApplication.processEvents()

    def iter_swap(self, first, last, callback=None):
        first.data, last.data = last.data, first.data

        if callback:
            callback()
            QtCore.QCoreApplication.processEvents()
```

```
class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(688, 509)
        self.lineEdit =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit.setGeometry(QtCore.QRect(21, 63,
251, 21))
        self.lineEdit.setObjectName("lineEdit")
        self.lineEdit_2 =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit_2.setGeometry(QtCore.QRect(21,
198, 251, 21))
        self.lineEdit_2.setObjectName("lineEdit_2")
        self.lineEdit_3 =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit_3.setGeometry(QtCore.QRect(21,
225, 251, 21))
        self.lineEdit_3.setObjectName("lineEdit_3")
        self.lineEdit_4 =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit_4.setGeometry(QtCore.QRect(21,
352, 251, 21))
        self.lineEdit_4.setObjectName("lineEdit_4")
        self.lineEdit_5 =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit_5.setGeometry(QtCore.QRect(21,
379, 251, 21))
        self.lineEdit_5.setObjectName("lineEdit_5")
        self.pushButton_3 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_3.setGeometry(QtCore.QRect(71,
90, 161, 24))
        self.pushButton_3.setObjectName("pushButton_3")
        self.pushButton_4 =
```

```
QtWidgets.QPushButton(parent=Form)
        self.pushButton_4.setGeometry(QtCore.QRect(70,
260, 161, 24))
        self.pushButton_4.setObjectName("pushButton_4")
        self.pushButton_5 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_5.setGeometry(QtCore.QRect(70,
410, 161, 24))
        self.pushButton_5.setObjectName("pushButton_5")
        self.textEdit =
QtWidgets.QTextEdit(parent=Form)
        self.textEdit.setGeometry(QtCore.QRect(301, 61,
341, 101))
        self.textEdit.setObjectName("textEdit")
        self.textEdit_2 =
QtWidgets.QTextEdit(parent=Form)
        self.textEdit_2.setGeometry(QtCore.QRect(301,
190, 346, 100))
        self.textEdit_2.setObjectName("textEdit_2")
        self.textEdit_3 =
QtWidgets.QTextEdit(parent=Form)
        self.textEdit_3.setGeometry(QtCore.QRect(301,
330, 348, 100))
        self.textEdit_3.setObjectName("textEdit_3")

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)


        self.linked_list = DoublyLinkedList()

        values = [random.randint(1, 100) for _ in
range(10)]
        for value in values:
            self.linked_list.append(value)
```

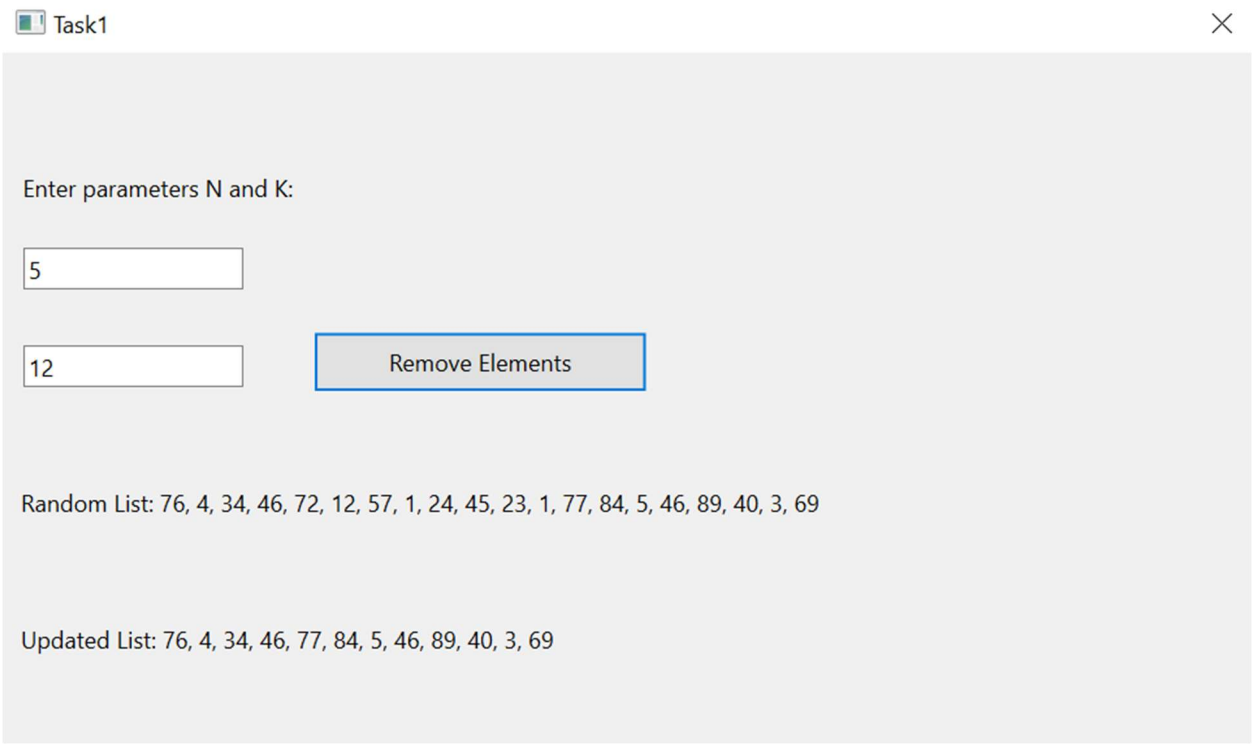
```
self.pushButton_3.clicked.connect(self.count_elements)

self.pushButton_4.clicked.connect(self.reverse_elements
)

self.pushButton_5.clicked.connect(self.iter_swap_elemen
ts)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.pushButton_3.setText(_translate("Form",
"Count the occurrence"))
        self.pushButton_4.setText(_translate("Form",
"Reverse a list"))
        self.pushButton_5.setText(_translate("Form",
"Iter swap a list"))

    def count_elements(self):
        value_to_count = int(self.lineEdit.text())
        count = self.linked_list.count(value_to_count)
        self.textEdit.clear()
        self.textEdit.append(f"The value
{value_to_count} appears {count} times in the linked
list.")

    def reverse_elements(self):
        start_idx = int(self.lineEdit_2.text())
        end_idx = int(self.lineEdit_3.text())

        def callback():
            self.textEdit_2.clear()
            self.textEdit_2.append(f"Elements reversed
```

```
in the specified range (indices {start_idx}-{end_idx})
in the linked list.")
            self.display_linked_list(self.textEdit_2)

        self.linked_list.reverse(start_idx, end_idx,
callback)

    def iter_swap_elements(self):
        index_1 = int(self.lineEdit_4.text())
        index_2 = int(self.lineEdit_5.text())

        def callback():
            self.textEdit_3.clear()
            self.textEdit_3.append(f"Elements at
indices {index_1} and {index_2} swapped in the linked
list.")
            self.display_linked_list(self.textEdit_3)


self.linked_list.iter_swap(self.linked_list.head,
self.linked_list.tail, callback)

    def display_linked_list(self, output_field):
        current = self.linked_list.head
        elements = []

        while current:
            elements.append(current.data)
            current = current.next

        output_field.append("Resulting Linked List: " +
", ".join(map(str, elements)))

if __name__ == "__main__":
    import sys
```

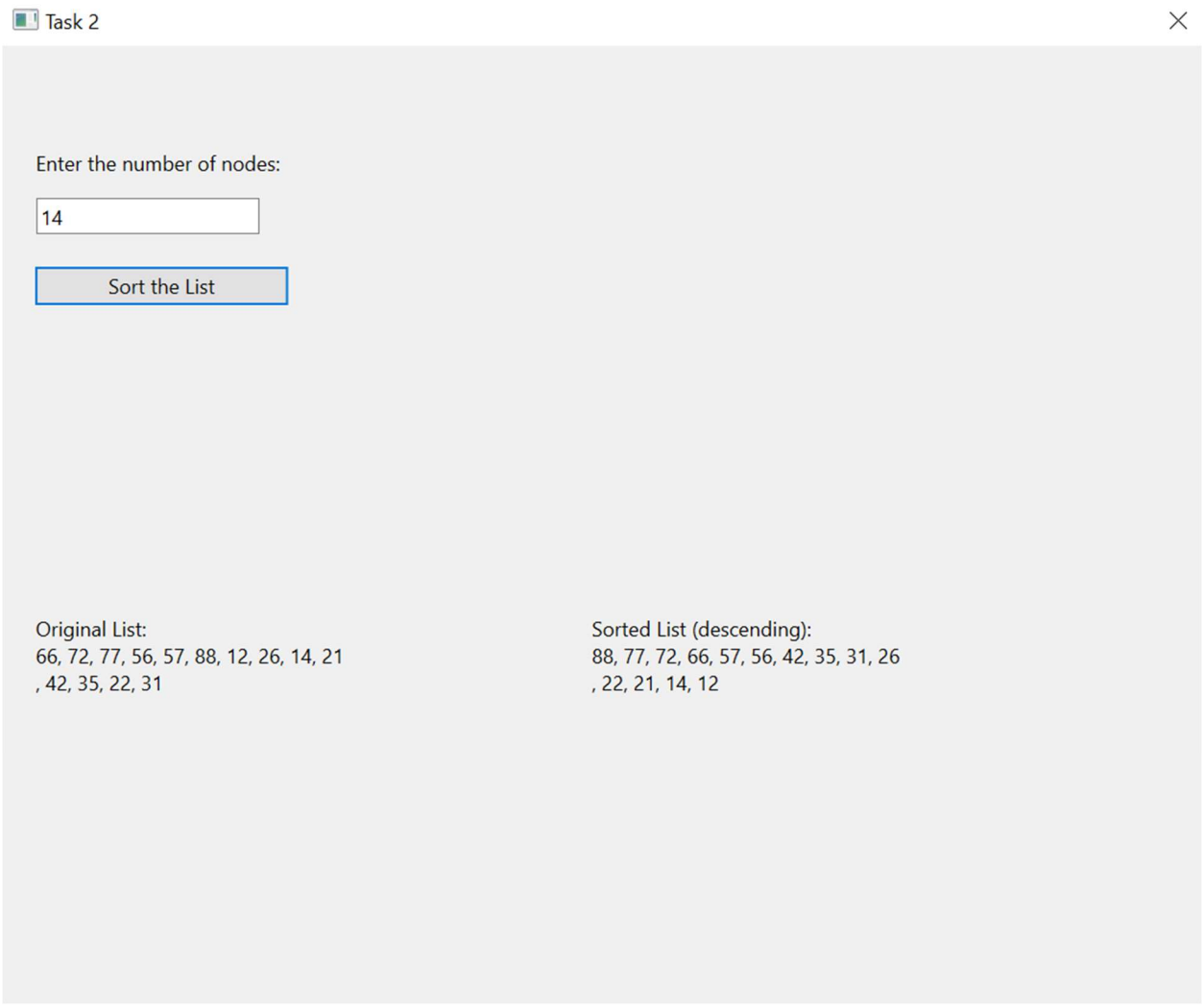
```
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec())
```

Скрін-шоти виконання завдань лабораторної роботи :

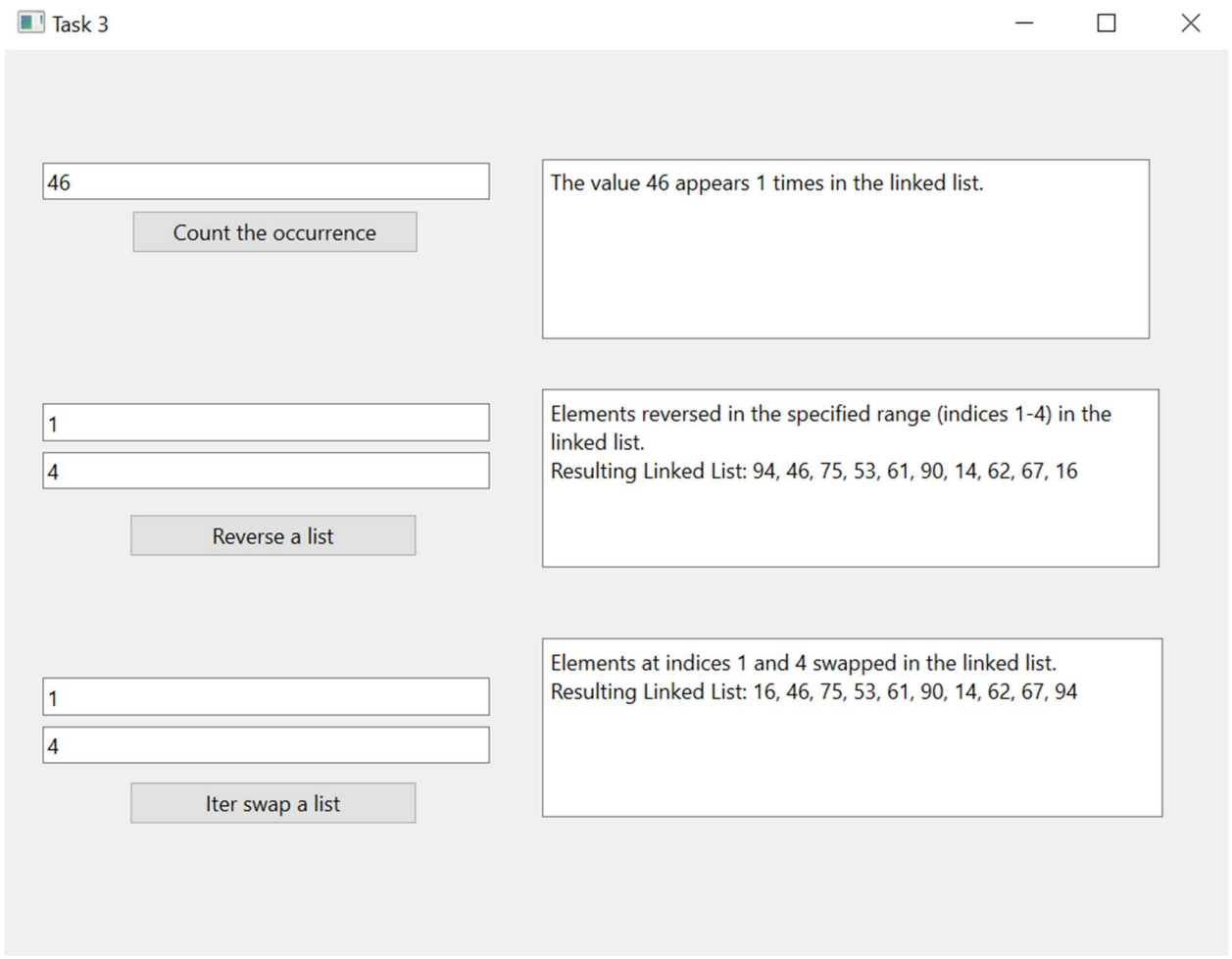
Завдання №1 :



Завдання №2 :


Task 2
Enter the number of nodes:
14
Sort the List
Original List:
66, 72, 77, 56, 57, 88, 12, 26, 14, 21
, 42, 35, 22, 31
Sorted List (descending):
88, 77, 72, 66, 57, 56, 42, 35, 31, 26
, 22, 21, 14, 12

Завдання №3 :


Task 3
46
Count the occurrence
The value 46 appears 1 times in the linked list.
1
4
Reverse a list
Elements reversed in the specified range (indices 1-4) in the linked list.
Resulting Linked List: 94, 46, 75, 53, 61, 90, 14, 62, 67, 16
1
4
Iter swap a list
Elements at indices 1 and 4 swapped in the linked list.
Resulting Linked List: 16, 46, 75, 53, 61, 90, 14, 62, 67, 94