Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Звіт

з лабораторної роботи № 6

«Побудування програмної системи з множини об’єктів, керованих повідомленнями»

Варіант 29

Виконала студентка ІМ-11, Пономаренко Маргарита Альбертівна
(шифр, прізвище, ім'я, по батькові)

Перевірив Порєв В. М.
(прізвище, ім'я, по батькові)

Київ 2022

Мета роботи: отримати вміння та навички використовувати засоби обміну інформацією та запрограмувати взаємодію незалежно працюючих програмних компонентів.

Завдання:

1. Створити у середовищі Qt Creator C++ проект з ім'ям Lab6.
2. Написати вихідні тексти усіх програм-компонентів згідно варіанту завдання.
3. Скомпілювати вихідні тексти і отримати виконувані файли програм.
4. Перевірити роботу програм. Налагодити взаємодію програм.
5. Проаналізувати та прокоментувати результати та вихідні тексти програм.
6. Оформити звіт.

Варіант 29

1	1. Користувач вводить значення **n, Min, Max** у діалоговому вікні. 2. Програма викликає програми Object2, 3 і виконує обмін повідомленнями з ними для передавання, отримання інформації.	1. Створює матрицю **n×n** цілих (**int**) чисел у діапазоні **Min – Max** 2. Показує числові значення у власному головному вікні 3. Записує дані в Clipboard Windows у текстовому форматі	1. Зчитує дані з Clipboard Windows 2. Відображає значення детермінанту матриці у власному головному вікні

Текст головного файлу (main.cpp)

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
  QApplication a(argc, argv);
  MainWindow w;
  w.move(10, 10);
  w.show();
  return a.exec();
}
```

Текст заголовку головного вікна (mainwindow.h)

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H
```

```
#include <QMainWindow>
#include <QProcess>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
  Q_OBJECT

public:
  MainWindow(QWidget *parent = nullptr);
  ~MainWindow();

private slots:

  void on_generateButton_pressed();

  void on_generateButton_released();

private:
  Ui::MainWindow *ui;
  QProcess *first;
  QProcess *second;
};
#endif // MAINWINDOW_H
```

Текст реалізації головного вікна (mainwindow.cpp)

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
  : QMainWindow(parent)
  , ui(new Ui::MainWindow)
{
  ui->setupUi(this);
  setWindowTitle("Lab 6");
  first = new QProcess(this);
  second = new QProcess(this);


}

MainWindow::~MainWindow()
{
  delete ui;
}

void MainWindow::on_generateButton_pressed()
```

```
{
  QStringList arguments;
  arguments << ui->numberValue->text() << ui->minValue->text() <<
ui->maxValue->text()<<QString::number(this->width())<<QString::number(this->x());
  qDebug()<<arguments;
  first->start("Object1.exe", arguments);
}


void MainWindow::on_generateButton_released()
{
  QStringList arguments;
  second->start("Object2.exe", arguments);
}
```

Текст головного файлу Object1(main.cpp)

```
#include "mainwindow.h"

#include <QApplication>
#include <QList>

int main(int argc, char *argv[])
{
  QApplication a(argc, argv);
  QList<QString> list;
  for(int i=1; i<argc; i++){
     list.append(argv[i]);
  }

  MainWindow w(list);
  if(list.size() == 5){
     w.move(list.at(3).toInt() + list.at(4).toInt() + 10, 10);
  }
  w.show();
  return a.exec();
}
```

Текст заголовку головного вікна Object1(mainwindow.h)

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QList>
#include <QMainWindow>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE
```

```
class MainWindow : public QMainWindow
{
  Q_OBJECT

public:
  MainWindow(QList<QString> list, QWidget *parent = nullptr);
  ~MainWindow();


private:
  Ui::MainWindow *ui;
  QString matrixContent;

  void generateMatrix(QList<QString> list);
  void copyMatrixToClipboard();
};
#endif // MAINWINDOW_H
```

Текст реалізації головного вікна Object1(mainwindow.cpp)

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

#include <QClipboard>
#include <QRandomGenerator>

MainWindow::MainWindow(QList<QString> list, QWidget *parent)
  : QMainWindow(parent)
  , ui(new Ui::MainWindow)
{
  ui->setupUi(this);
  setWindowTitle("Object 1");
  qDebug()<<"List size: " <<list.size();

  generateMatrix(list);
  copyMatrixToClipboard();
}

MainWindow::~MainWindow()
{
  delete ui;
}

void MainWindow::generateMatrix(QList<QString> list)
{
  if(list.size()<3){
     return;
  }

  int matrixSize = list.at(0).toInt();
  int minValue = list.at(1).toInt();
```

```
  int maxValue = list.at(2).toInt();
  qDebug()<<"Size: "<<matrixSize<<" Min: "<<minValue<<" Max: "<<maxValue;

  QRandomGenerator generator = QRandomGenerator::securelySeeded();
  int value;
  for(int i=0; i<matrixSize; i++){
     for(int j=0; j<matrixSize; j++){
        value = generator.bounded(minValue, maxValue);
        matrixContent.append(QString::number(value) + "\t");
     }
     matrixContent.append("\n");
  }
  ui->textEdit->setText(matrixContent);
}

void MainWindow::copyMatrixToClipboard()
{
  QClipboard *clipboard = QGuiApplication::clipboard();
  clipboard->setText(matrixContent);
}
```

Текст головного файлу Object2(main.cpp)

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
  QApplication a(argc, argv);
  MainWindow w;
  w.move(w.width() * 2 + 50, 10);
  w.show();
  return a.exec();
}
```

Текст заголовку головного вікна Object2(mainwindow.h)

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
  Q_OBJECT
```

```
public:
  MainWindow(QWidget *parent = nullptr);
  ~MainWindow();

private:
  Ui::MainWindow *ui;
  QString matrixContentRead;
  void readFromClipboard();
  int calculateDeterminant(QVector <QVector <int> > matrix, int matrixSize);

};
#endif // MAINWINDOW_H
```

Текст реалізації головного вікна Object2(mainwindow.cpp)

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

#include <QClipboard>

MainWindow::MainWindow(QWidget *parent)
  : QMainWindow(parent)
  , ui(new Ui::MainWindow)
{
  ui->setupUi(this);
  setWindowTitle("Object 2");
  readFromClipboard();
}

MainWindow::~MainWindow()
{
  delete ui;
}

void MainWindow::readFromClipboard()
{
  QClipboard *clipboard = QGuiApplication::clipboard();
  matrixContentRead = clipboard->text();
  //matrixContentRead = "3\t5\t6\n4\t2\t3\n5\t7\t5\n";
  ui->label->setText(matrixContentRead );

  int matrixSize = matrixContentRead.count('\n');  //розрахунок matrixSize (рахує правильно)
  ui->label->setText(QString::number(matrixSize));

  QList<QString> splitByChar = matrixContentRead.split("\n");
  matrixContentRead = "";
  for(int n=0; n<splitByChar.length(); n++){
     ui->lineEdit->insert(splitByChar.at(n));
     matrixContentRead.append(splitByChar.at(n));
  }
```

```
    ui->label->setText(QString::number(splitByChar.length()));
    qDebug()<<"QString list: "<<splitByChar;

    QVector<int> matrixContent; //вектор чисел без знаку табуляції
    QString tempNum = "";
    for(auto slotN:splitByChar){
        for(auto charact:slotN){
            if(charact == '\t'){
                matrixContent.append(tempNum.toInt());
                tempNum = "";
            }
            if(charact != '\t'){
                tempNum += charact;
            }
        }
    }
    for(auto item:matrixContent){
        ui->lineEdit->insert(QString::number(item));
    }

    QVector <QVector <int> > secondMatrixContent; //двомірний вектор чисел
    int tempCounter = 0;
    for(int i=0; i<matrixSize; i++){
        QVector<int> tempVector;
        for(int j=0; j<matrixSize; j++){
            tempVector.push_back(matrixContent.at(tempCounter));
            tempCounter++;
        }
        secondMatrixContent.push_back(tempVector);
    }

    ui->lineEdit->setText("");
    for(auto item:secondMatrixContent){
        for(auto subItem:item){
            ui->lineEdit->insert(QString::number(subItem) + " ");
        }
         ui->lineEdit->insert("\n");
    }

    int determinant = calculateDeterminant(secondMatrixContent, matrixSize);
    ui->label->setText("Determinant of matrix is : " + QString::number(determinant));

}

int MainWindow::calculateDeterminant(QVector<QVector<int> > matrix, int matrixSize)
{
    int det = 0;
    QVector<QVector<int> > submatrix(10, QVector <int>(10));
    if (matrixSize == 2){
        return ((matrix[0][0] * matrix[1][1]) - (matrix[1][0] * matrix[0][1]));
    }
    else {
```

```
    for (int x = 0; x < matrixSize; x++) {
        int subi = 0;
        for (int i = 1; i < matrixSize; i++) {
            int subj = 0;
            for (int j = 0; j < matrixSize; j++) {
                if (j == x)
                    continue;
                submatrix[subi][subj] = matrix[i][j];
                subj++;
            }
            subi++;
        }
        det = det + (pow(-1, x) * matrix[0][x] * calculateDeterminant( submatrix, matrixSize - 1 ));
    }
  }
  return det;
}
```

Результати тестування програми

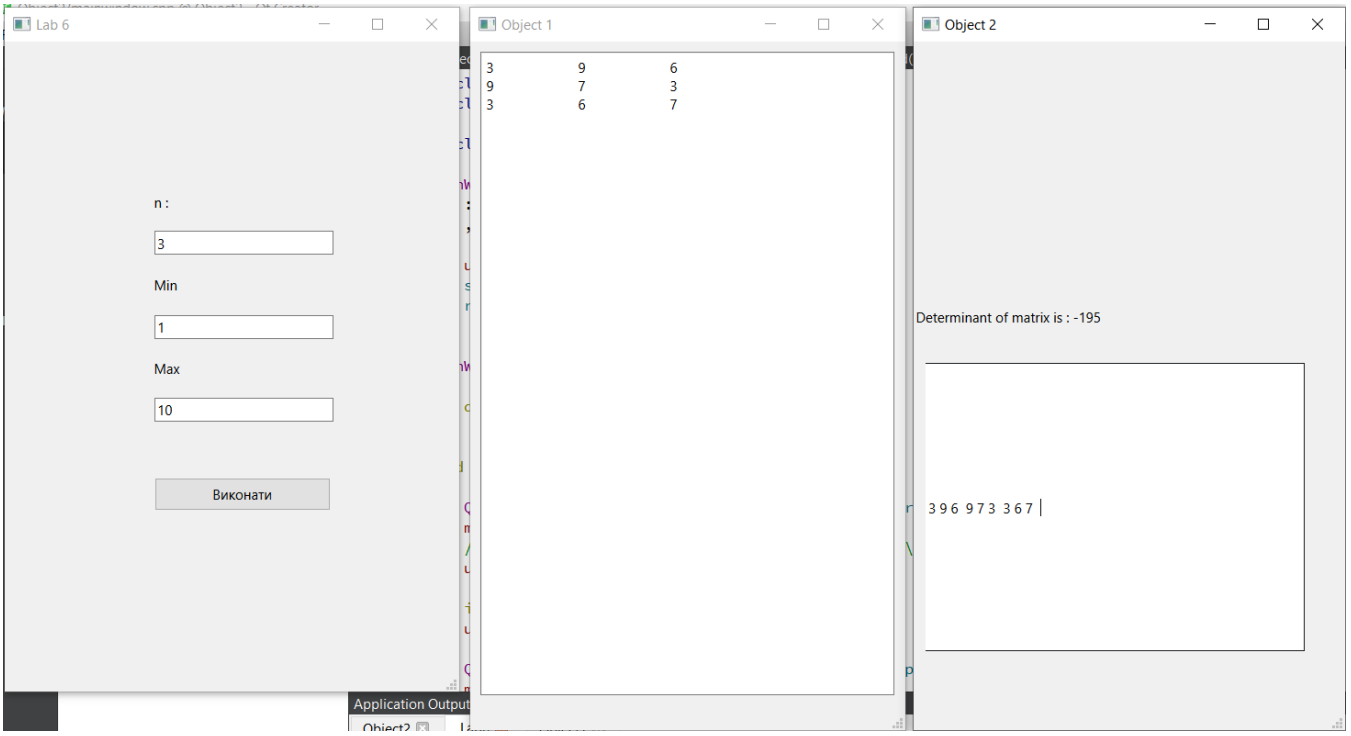


Рис 1.1: Демонстрація запуску програм Object1 та Object2

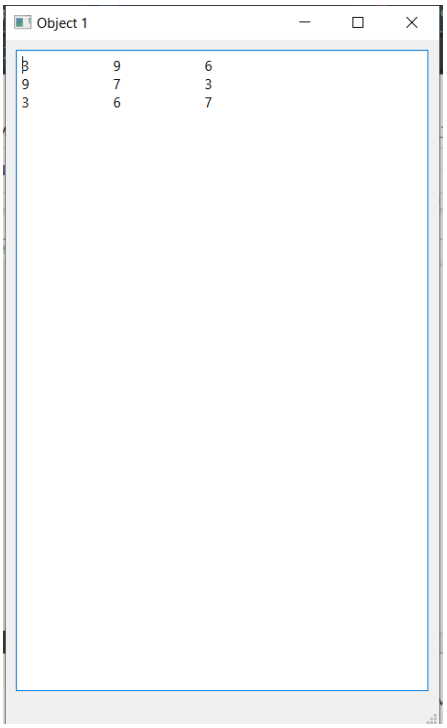


Рис 1.2: Демонстрація роботи Object1, а саме виведення згенерованої матриці

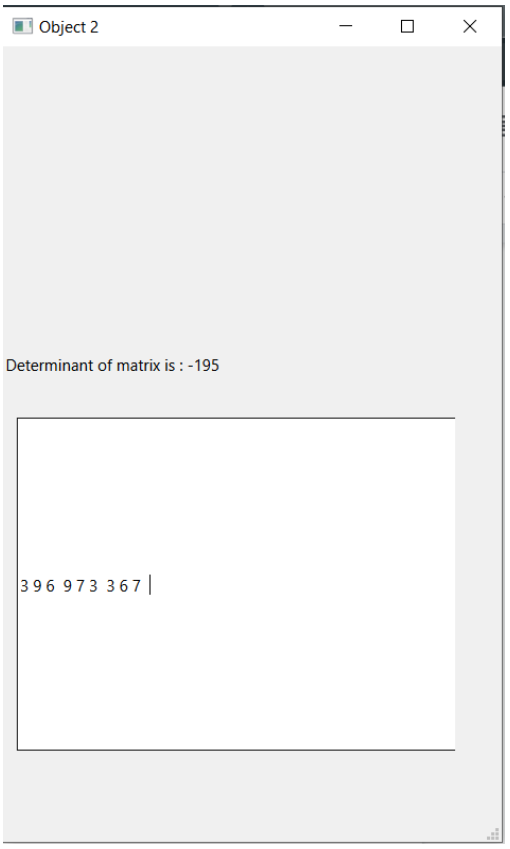
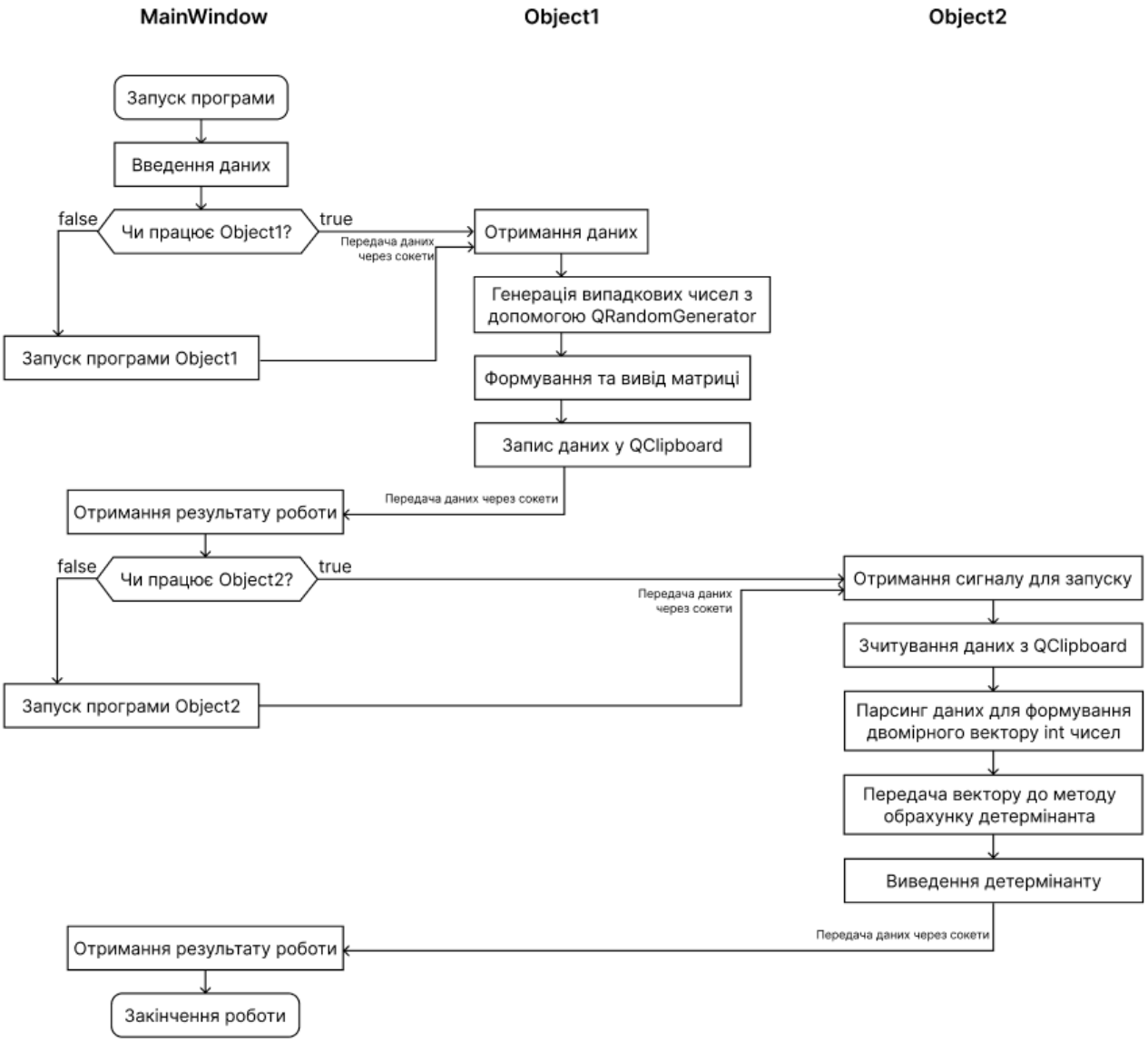


Рис 1.3: Демонстрація роботи Object2, а саме виведення значення детермінанта та додатково виведення вектору int матриці, що була отримана з QClipboard

Схема відправлення та отримання повідомлень



Висновок

У цій лабораторній роботи отримано вміння та навички використовувати засоби обміну інформацією та запрограмовано взаємодію незалежно працюючих програмних компонентів Lab6, Object1 та Object2. Також випробувано на практиці вміння запису та читання з QClipboard.