Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій

Кафедра ПЗ

Звіт

до лабораторної роботи №12

на тему «Виняткові ситуації в мові програмування С++»

з дисципліни “Об’єктно-орієнтоване програмування

Виконав:

студент групи ПЗ-11

Ясногородський Н.В

Перевірила:

доц.
Коротєєва Т.О.

Львів

2022

Лабораторна робота №12

Виняткові ситуації в мові програмування С++

Тема: Виняткові ситуації в мові програмування С++.

Мета: Ознайомитися з синтаксисом та принципами використання винятків, навчитися передбачати виняткові ситуації, які можуть виникнути в процесі роботи програмного забезпечення, а також навчитися їх перехоплювати та опрацьовувати.

Теоретичні відомості

В основі обробки виняткових ситуацій у мові С++ лежать три ключових слова: try, catch і throw.

Якщо програміст підозрює, що визначений фрагмент програми може спровокувати помилку, він повинний занурити цю частину коду в блок try. Необхідно мати на увазі, що зміст помилки (за винятком стандартних ситуацій) визначає сам програміст. Це значить, що програміст може задати будь-яку умову, що приведе до створення виняткової ситуації. Після цього необхідно вказати, у яких умовах варто генерувати виняткову ситуацію. Для цієї мети призначене ключове слово throw. І нарешті, виняткову ситуацію потрібно перехопити й обробити в блоці catch. Ось як виглядає ця конструкція.

```
try
{
// Тіло блоку try
if(умова) throw виняткова ситуація
}
catch(тип1 аргумент)
{
// Тіло блоку catch
}
catch(тип2 аргумент)
{
// Тіло блоку catch
}
.
.
.
catch(тип N аргумент)
{
// Тіло блоку catch
 }
```

Розмір блоку try не обмежений. У нього можна занурити як один оператор, так і цілу програму. Один блок try можна зв'язати з довільною кількістю блоків catch. Оскільки кожен блок catch відповідає окремому типу виняткової ситуації, програма сама визначить, який з них виконати. У цьому випадку інші блоки catch не виконуються. Кожен блок catch має аргумент, що приймає визначене значення. Цей аргумент може бути об'єктом будь-якого типу.

Якщо програма виконана правильно й у блоці try не виникло жодної виняткової ситуації, усі блоки catch будуть зігноровані. Якщо в програмі виникла подія, що програміст вважає небажаним, оператор throw генерує виняткову ситуацію. Для цього оператор throw повинний знаходитися усередині блоку try або усередині функції, викликуваної усередині блоку try.

Індивідуальне завдання

Варіант 5:

5. Реалізувати програму для обчислення виразу

(a11 + ...+ a1n)*√x + (a21 +...+a2m)*log(100-x) + (a31 + ...+a3k)/(x-10)

Коефіцієнти зчитуються з клавіатури. Роботу з виразом потрібно здійснювати за допомогою класу Vyraz. Програма повинна перехоплювати та опрацьовувати такі виняткові ситуації: а) ділення на нуль, б) помилкове введення користувачем літерного символу замість числа при введенні коефіцієнтів, в) переповнення, г)вихід за межі масиву коефіцієнтів, д) ще дві виняткові ситуації передбачити самостійно.

Код програми

Файл main.cpp:

```
#include "mainwindow.h"

#include <QApplication>

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

Файл widget.cpp:

```
#include "widget.h"

#include <QApplication>
#include <QFile>
#include <QGridLayout>
#include <QMessageBox>
#include <QTextStream>
#include <algorithm>
#include <iostream>
#include <random>
#include <sstream>

double get_random_double() {
  std::random_device rd;
  std::default_random_engine eng(rd());
  std::uniform_real_distribution<double> distr(0, 100);
  return distr(eng);
}

void Widget::on_set() {
  auto n = this->n_input->toPlainText().toInt();
  auto m = this->m_input->toPlainText().toInt();
  auto k = this->k_input->toPlainText().toInt();
  this->vyraz->coff_n.resize(n);
```

```
  this->vyraz->coff_m.resize(m);
  this->vyraz->coff_k.resize(k);
  this->coff_table->setRowCount(3);
  this->coff_table->setColumnCount(std::max({n, m, k}));

  for (auto i = 0; i < this->coff_table->rowCount(); i++)
    for (auto j = 0; j < this->coff_table->columnCount(); j++)
      this->coff_table->setItem(
          i, j, new QTableWidgetItem(QString::number(get_random_double())));
}

void Widget::on_calculate() {
  try {
    std::vector<double>* temp;
    for (auto i = 0; i < this->coff_table->rowCount(); i++) {
      switch (i) {
        case 0:
          temp = &this->vyraz->coff_n;
          break;
        case 1:
          temp = &this->vyraz->coff_m;
          break;
        case 2:
          temp = &this->vyraz->coff_k;
          break;
      }
      for (auto j = 0; j < this->coff_table->columnCount(); j++) {
        if (temp->size() < j + 1) throw "Out of bounds assignment";

        (*temp)[j] = this->coff_table->item(i, j)->text().toDouble();
      }
    }
    bool ok;
    auto x = this->x_input->toPlainText().toDouble(&ok);
    if (!ok) throw "X should be a number";

    this->value_output->setText(QString::number(this->vyraz->calculate(x)));
  } catch (const char* e) {
    QMessageBox error;
    error.setText(QString(e));
    error.exec();
  } catch (...) {
    QMessageBox error;
    error.setText(QString::fromStdString("Unknown Error :("));
    error.exec();
  }
}

Widget::Widget(QWidget* parent) : QWidget(parent) {
  this->vyraz = new Vyraz;
  auto main_layout = new QGridLayout;
  this->start_btn = new QPushButton("Calculate");
  this->set_btn = new QPushButton("Set");
  this->n_input = new QTextEdit("3");
  this->m_input = new QTextEdit("3");
  this->k_input = new QTextEdit("3");
  this->x_input = new QTextEdit("11");
  this->value_output = new QTextEdit;
  this->value_output->setReadOnly(true);
  this->coff_table = new QTableWidget;

  main_layout->addWidget(this->n_input, 0, 0);
  main_layout->addWidget(this->m_input, 0, 1);
  main_layout->addWidget(this->k_input, 0, 2);
  main_layout->addWidget(this->x_input, 0, 3);
  main_layout->addWidget(this->set_btn, 1, 0);
  main_layout->addWidget(this->coff_table, 2, 0, 2, 4);
  main_layout->addWidget(this->start_btn, 4, 0);
```

```
  main_layout->addWidget(this->value_output, 4, 2);

  connect(this->start_btn, &QPushButton::released, this, &Widget::on_calculate);
  connect(this->set_btn, &QPushButton::released, this, &Widget::on_set);

  setLayout(main_layout);
}
```

Файл vyraz.cpp:

```
#include "vyraz.h"

#include "cmath"
#include "numeric"

double Vyraz::calculate(double x) {
  if (x < 0) throw "Sqrt of number less then 0";
  if (x >= 100) throw "Log10 of number less or equal than 0";
  if (x == 10) throw "Division by 0";

  return (std::accumulate(coff_n.begin(), coff_n.end(), 0) * sqrt(x)) +
         (std::accumulate(coff_m.begin(), coff_m.end(), 0) * log10(100 - x)) +
         (std::accumulate(coff_k.begin(), coff_k.end(), 0) / (x - 10));
}
```

Результат виконання програми:

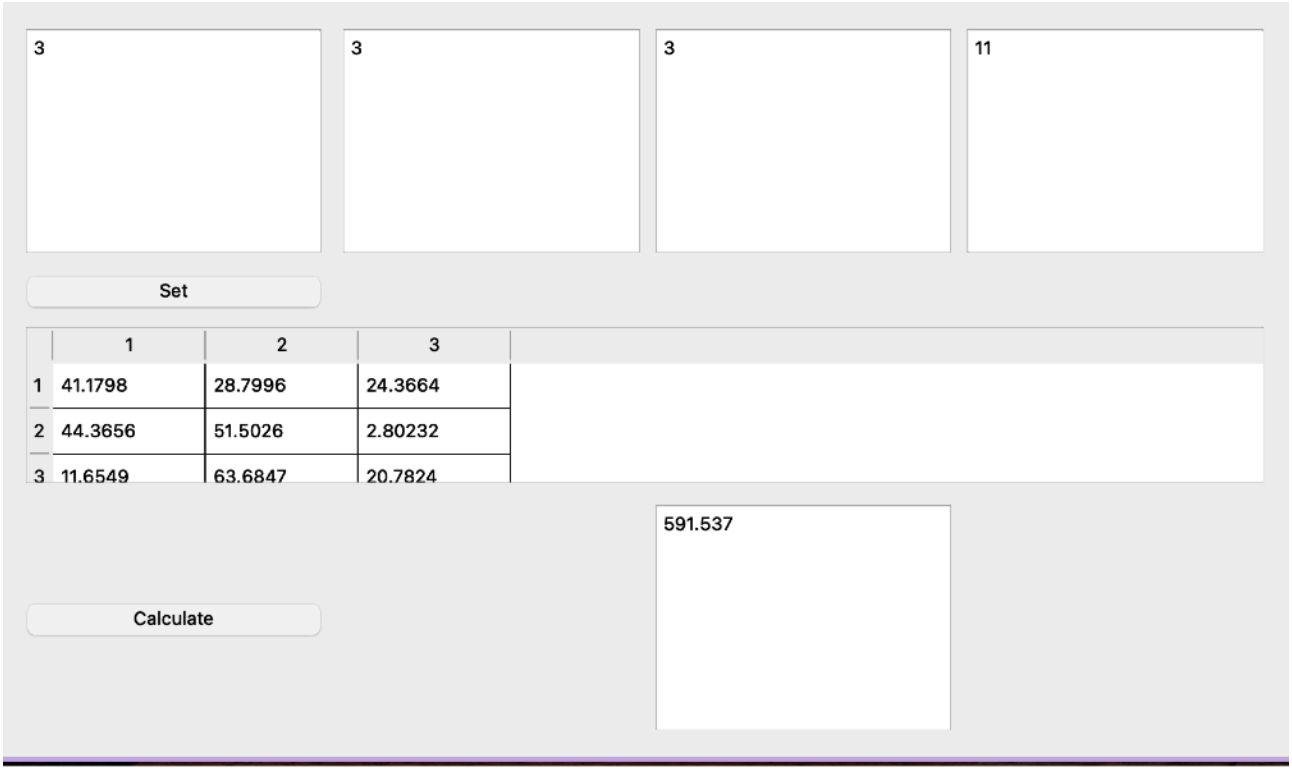

Рис. 1

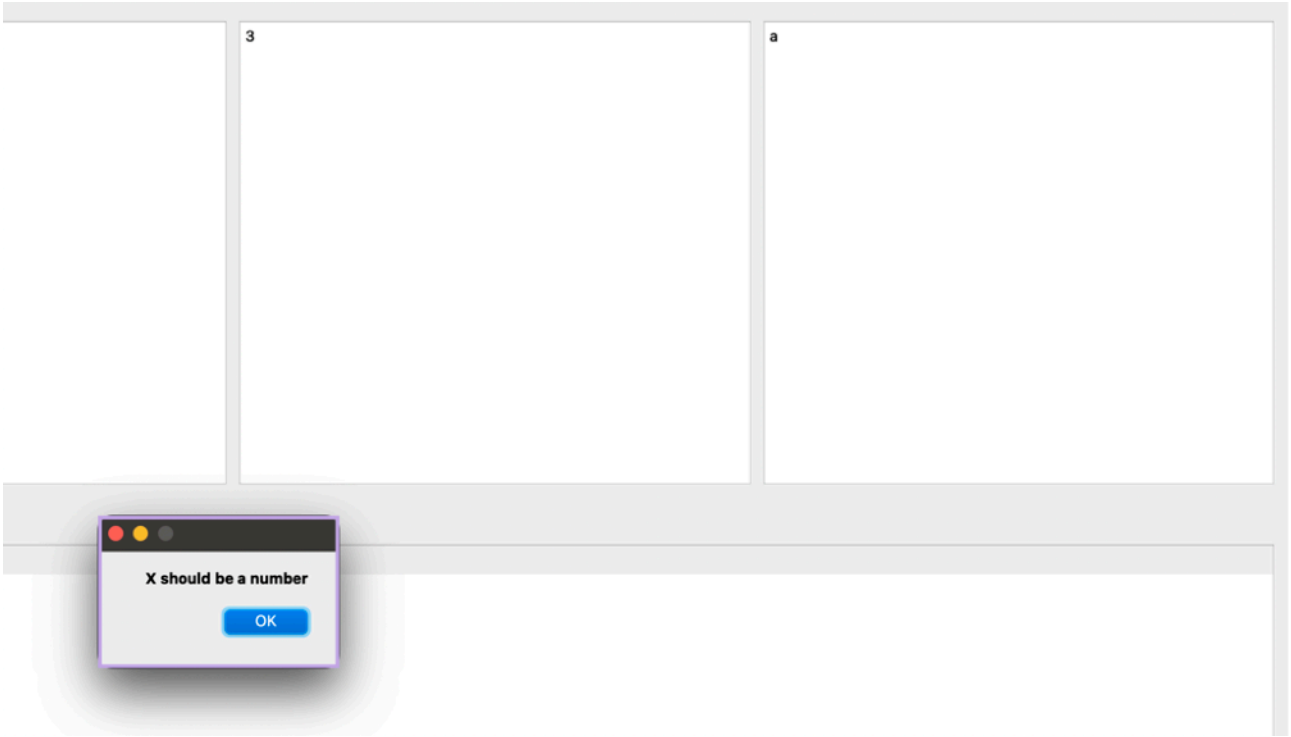

Рис. 2

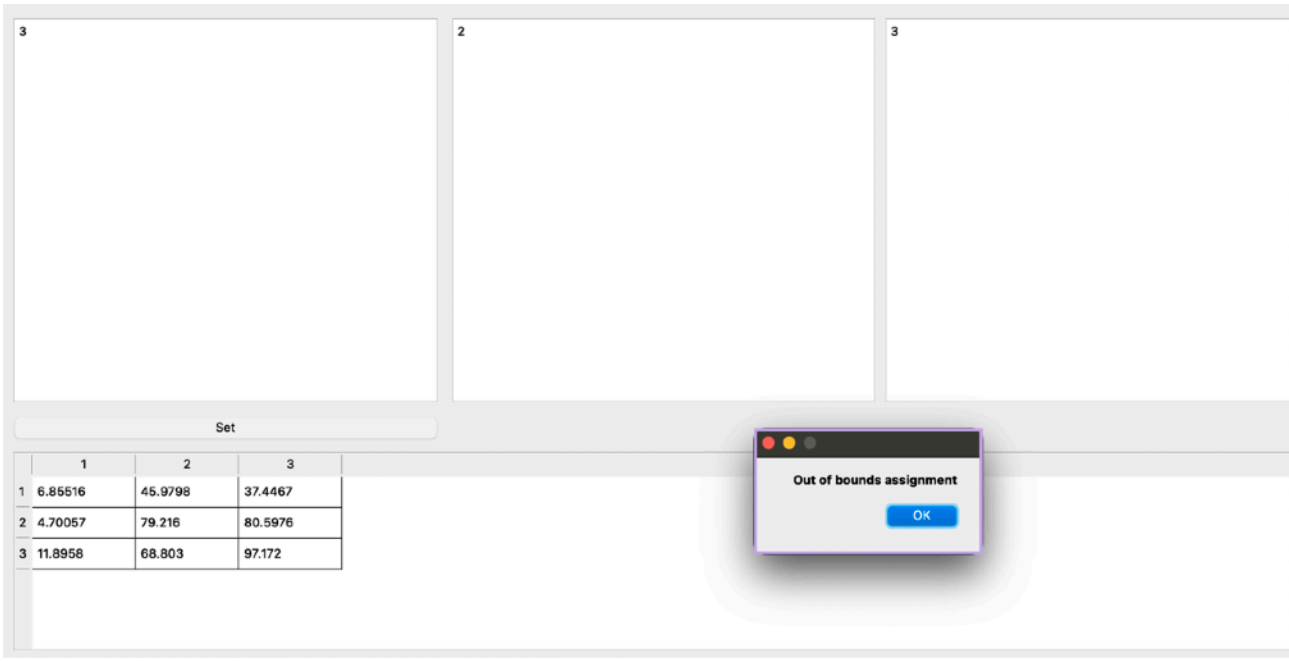

Рис. 3

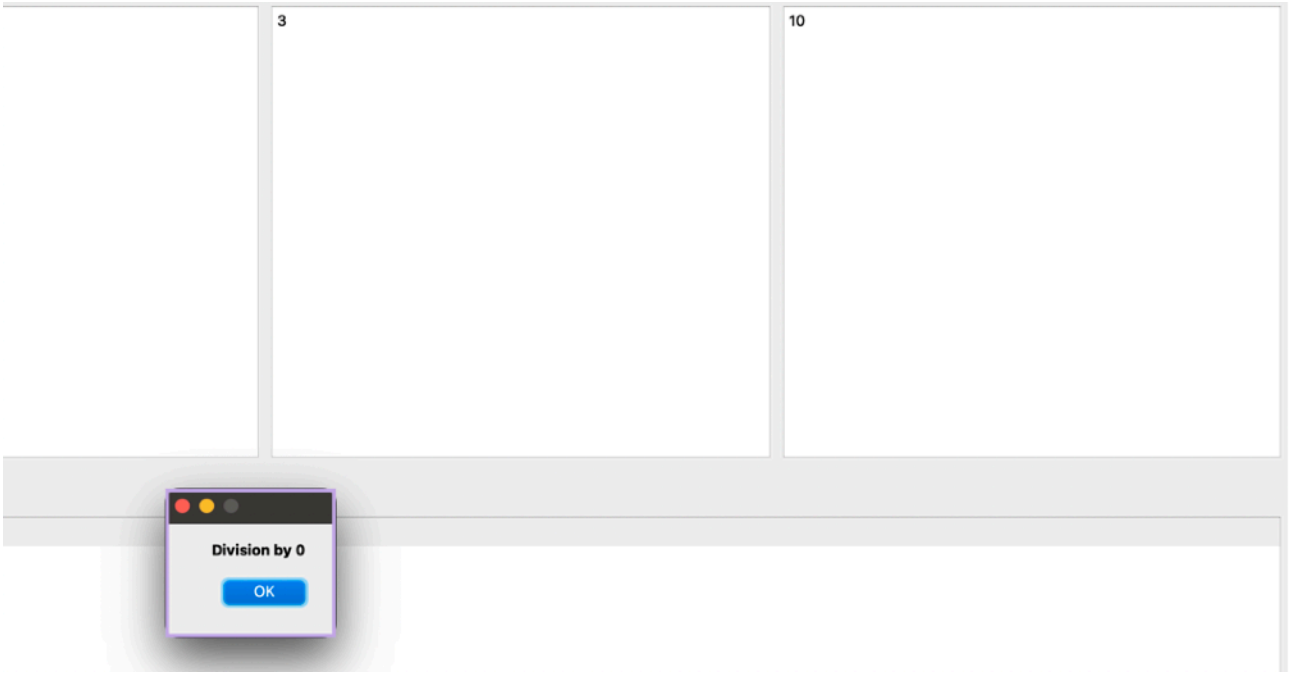

Рис.4

Висновок: Виконуючи лабораторну роботу №12 я ознайомився з синтаксисом та принципами використання винятків, навчився передбачати виняткові ситуації, які можуть виникнути в процесі роботи програмного забезпечення, а також навчився їх перехоплювати та опрацьовувати.