**НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ** **УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ** **«МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ** **“СИНЕРГИЯ”»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Факультет/Институт** | | |  | Информационных технологий | |
|  | | |  | (наименование факультета/ Института) | |
| **Направление/специальность** | |  | 09.02.07 Информационные системы и программирование | | |
| **подготовки:** | | |  | (код и наименование направления /специальности подготовки) | |
| **Форма обучения:** | | |  | Очная | |
|  | | |  | (очная, очно-заочная, заочная) | |
|  | | |  |  | |

**Отчет по лабораторному практикуму №4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **на тему** |  | **Анализ и обеспечение обработки исключительных ситуаций** | | |
|  |  | (наименование темы) | | |
|  |  |  | | |
| **по дисциплине** | | |  | Тестирование информационных систем |
|  | | |  | (наименование дисциплины) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Обучающийся** |  | Белозерцев Дмитрий Сергеевич |  |  |
|  |  | (ФИО) |  | (подпись) |
| **Группа** |  | ДКИП-311 |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Преподаватель** |  | **Авдеенков Владимир Александрович** |  |  |
|  |  | (ФИО) |  | (подпись) |

**Москва 2024**

**Лабораторная работа № 4: «Анализ и обеспечение обработки исключительных ситуаций»:**

**Цель:**

1. Получение навыков анализа и обеспечения обработки исключительных ситуаций.

**Задание 1:**

Код:

#include <iostream>

#include <stdexcept>

#include <ctime>

class YearError : public std::runtime\_error {

public:

explicit YearError(const std::string& message) : std::runtime\_error(message) {}

};

int calculate\_age(int year\_of\_birth) {

int current\_year = std::time(nullptr) / (365 \* 24 \* 60 \* 60) + 1970; // Получаем текущий год

if (year\_of\_birth < 0) {

throw YearError("Год рождения не может быть отрицательным.");

}

else if (year\_of\_birth > current\_year) {

throw YearError("Год рождения не может быть больше текущего года.");

}

return current\_year - year\_of\_birth;

}

int main() {

int year\_of\_birth;

while (true) {

try {

std::cout << "Введите ваш год рождения: ";

std::cin >> year\_of\_birth;

if (std::cin.fail()) {

std::cin.clear(); // Очищаем флаг ошибки

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n'); // Игнорируем некорректный ввод

throw std::invalid\_argument("Ошибка: Введите корректное число.");

}

int age = calculate\_age(year\_of\_birth);

std::cout << "Ваш возраст: " << age << " лет." << std::endl;

break; // Выход из цикла при успешном вводе

}

catch (const std::invalid\_argument& e) {

std::cout << e.what() << std::endl;

}

catch (const YearError& e) {

std::cout << e.what() << std::endl;

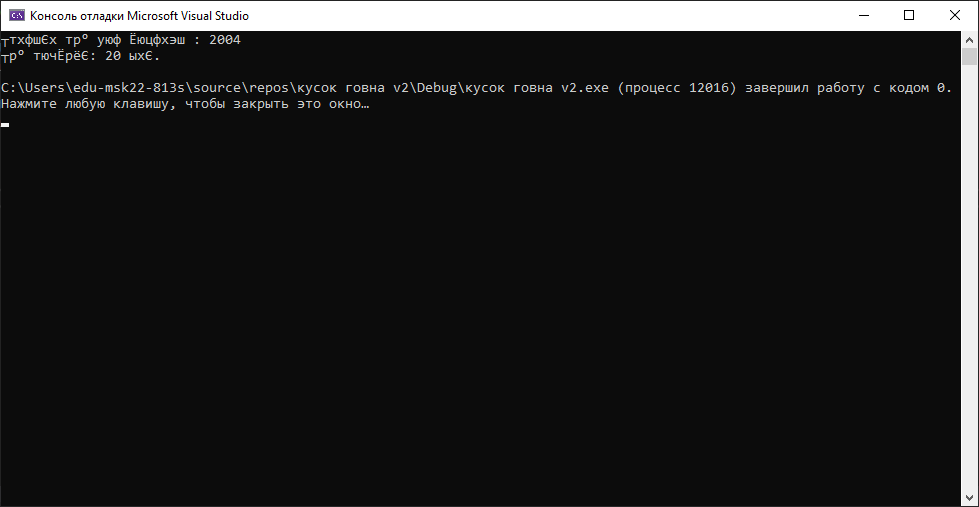
}

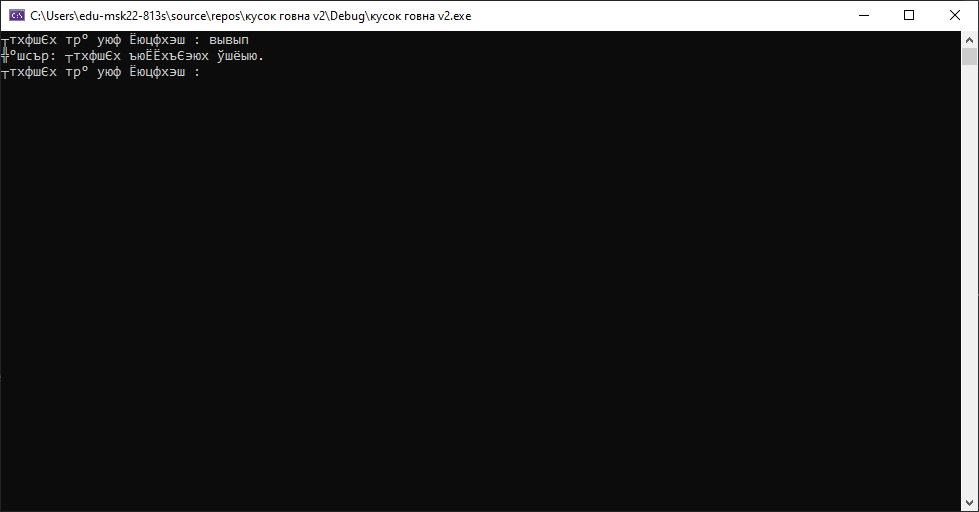
}

return 0;

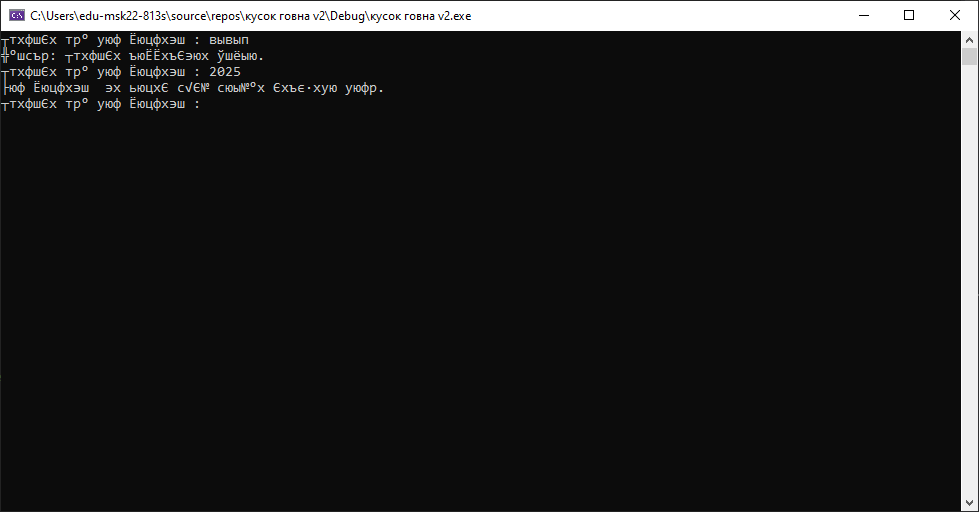
}

Работа программы:

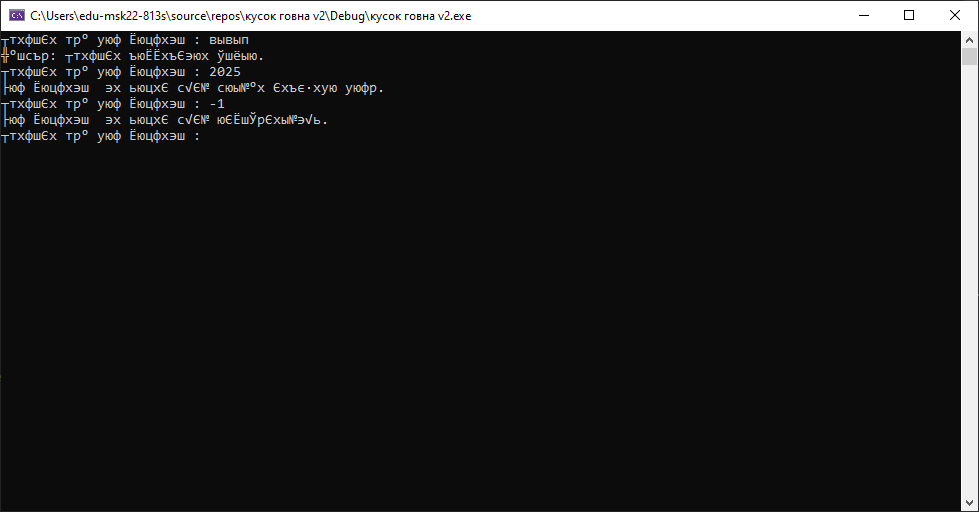
Результат правильной работы программы  


Проверка на корректность вводимых данных (Ошибка: Введите корректное число)  


Проверка, что год не может быть больше текущего



Проверка на отрицательный год рождения



**Задание 2:**

Код:

#include <iostream>

#include <cmath>

#include <stdexcept>

const double PI = 3.14159265358979323846;

double calculate(double A, double B, double C) {

if (A < -1.0 || A > 1.0) {

throw std::domain\_error("Ошибка: A должно быть в диапазоне [-1, 1].");

}

if (B < -1.0 || B > 1.0) {

throw std::domain\_error("Ошибка: B должно быть в диапазоне [-1, 1].");

}

if (tan(PI \* C) + 1 == 0) {

throw std::domain\_error("Ошибка: tan(PI \* C) + 1 не должно быть равно нулю.");

}

return acos(A) + sqrt(sin(PI \* B)) / (tan(PI \* C) + 1);

}

int main() {

double A, B, C;

std::cout << "Введите значение A (-1 <= A <= 1): ";

std::cin >> A;

std::cout << "Введите значение B (-1 <= B <= 1): ";

std::cin >> B;

std::cout << "Введите значение C: ";

std::cin >> C;

try {

double result = calculate(A, B, C);

std::cout << "Результат: " << result << std::endl;

}

catch (const std::domain\_error& e) {

std::cerr << e.what() << std::endl;

}

catch (const std::exception& e) {

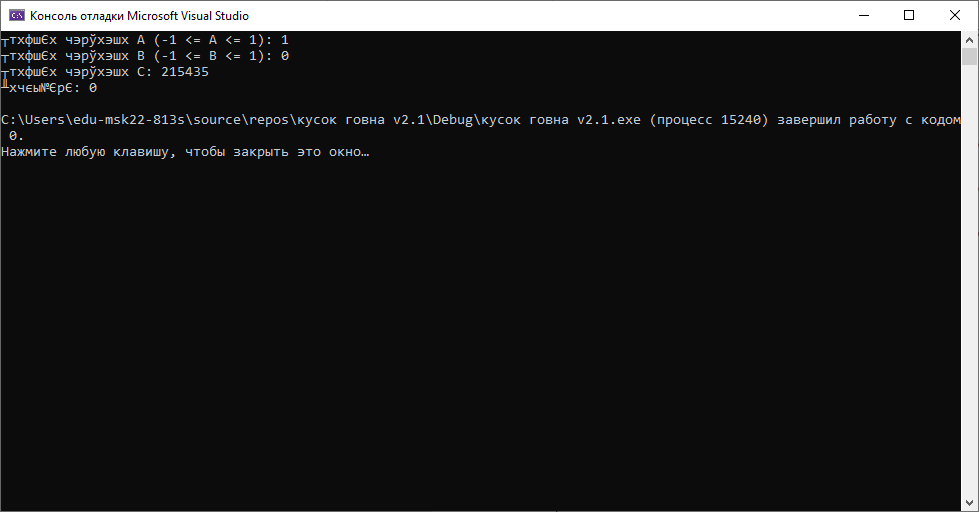
std::cerr << "Общая ошибка: " << e.what() << std::endl;

}

return 0;

}

Результат правильной работы программы



Исключение выхода за пределы  
