|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет  имени Н. Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н. Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | «Информатика и системы управления» (ИУ) |

|  |  |
| --- | --- |
| КАФЕДРА | «Информационная безопасность» (ИУ8) |

Лабораторная работа № 6

ПО КУРСУ

«Алгоритмические языки»

на тему «**Исключения**»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ8-23 |  |  |  | Д.С. Афанасьев |
|  | (Группа) |  |  |  | (И. О. Фамилия) |
|  |  |  |  |  |  |
| Преподаватель: |  |  |  |  | М. В. Малахов |
|  |  |  |  |  | (И.О. Фамилия) |

2024

Введение

Условия для 4 варианта

Реализовать класс «Простое число». Поля: значение числа. Конструкторы: конструктор для инициализации полей. При необходимости добавить свои функции. При недопустимых параметрах объекта (число непростое) создается исключение. Для исключения создать класс, производный от invalid\_argument. Дополнительно созданный класс для исключения кроме наследуемых полей (необходимо их также использовать) содержит поля класса «Простое число» для недопустимого объекта и метод для печати всех данных. Продемонстрировать работу с объектами в случае возникновения исключения и его обработки и без возникновения исключения.

Основная часть

Исходный текст программы:

#include <iostream>

#include <string>

class PrimeNumberException : public std::invalid\_argument {

protected:

  std::string message;

public:

  PrimeNumberException(const std::string& message, const std::string msg = "") : std::invalid\_argument(msg) {

*this*->message = message;

  }

  std::string getMessage() const {

    return message;

  }

};

class PrimeNumber {

protected:

  int value;

public:

*// PrimeNumber() = delete;*

  PrimeNumber(int value) {

    for (int i = 2; i \* i <= value; ++i) {

      if (value % i == 0) {

        throw PrimeNumberException("The number " + std::to\_string(value) + " is not a prime number");

      }

    }

*this*->value = value;

  }

  int getValue() const  {

    return value;

  }

};

std::ostream& operator<<(std::ostream& out, const PrimeNumber& primeNumber) {

  out << primeNumber.getValue() << std::endl;

  return out;

}

*//  g++ source/lab6/lab6.cpp -o build/lab6*

int main() {

  try {

    PrimeNumber p1(11);

    std::cout << p1 << std::endl;

    PrimeNumber p2(10);

    std::cout << p2 << std::endl;

  } catch (const PrimeNumberException& ex) {

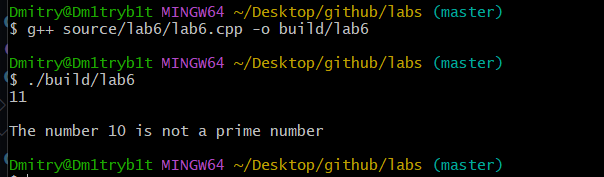
    std::cout << ex.getMessage() << std::endl;

  }

  return 0;

}

Снимки выполнения работы программы



Заключение

Задачи лабораторной работы были решены, результаты проверены. Изучена на практике работа с исключениями в языке C++.