|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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

КАФЕДРА ИУ8 “КОМПЬЮТЕРНАЯ БЕЗОПАСНОСТЬ”

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6**

Студент Лустин Павел Евгеньевич

*(*Фамилия И.О)

Группа\_\_ИУ8-24\_\_\_\_\_\_\_\_

Студент  **\_\_\_\_\_\_8.05\_\_\_\_\_\_\_\_\_**  Лустин \_\_ \_\_

*дата (*Фамилия И.О)

Преподаватель **\_\_\_\_\_\_8.05\_\_\_\_\_\_\_**  Барыкин Дмитрий

*дата (*Фамилия И.О)*.*

*2024 г.*

Цель работы

Цель работы состоит в изучении использования исключений в классе и их обработке

Условие задачи(вариант 5)

Реализовать класс «Массив – последовательность Фибоначчи». Поля: указатель на массив и его размерность. Конструкторы: конструктор для инициализации полей (через параметры передается обычный массив). При необходимости добавить свои функции. При недопустимых параметрах объекта создается исключение (массив не является последовательностью Фибоначчи). Для исключения создать класс, производный от invalid\_argument. Дополнительно созданный класс для исключения кроме наследуемых полей (необходимо их также использовать) содержит поля класса «Массив – последовательность Фибоначчи» для недопустимого объекта и метод для печати всех данных. Продемонстрировать работу с объектами в случае возникновения исключения и его обработки и без возникновения исключения.

Текст программы с комментариями

#include <iostream>

using namespace std;

class Exeption :public std::invalid\_argument //Класс для исключения, производный от invalid\_argument

{

public:

int\* p;

int n;

Exeption(const char\* msg, int\* p, int n) :invalid\_argument(msg) //Конструктор с заданными начальными значениями

{

this->p = p;

this->n = n;

}

void print() //Функция для печати всех данных

{

for (int i = 0; i < n; ++i)

{

cout << p[i] << " ";

}

}

};

class Fibonacci //Класс «Массив – последовательность Фибоначчи»

{

public:

int \*p; //Указатель на массив

int n; //Размерность массива

Fibonacci() //Пустой конструктор

{

p = nullptr;

n = 0;

}

Fibonacci(int\* p, int n) //Конструктор с начальными значениями передаваемого массива

{

for(int i = 0; i < n-2; ++i)

{

if (p[i] + p[i + 1] != p[i + 2]) //Цикл, который определяет, является ли массив последовательностью Фибоначчи или нет

{

throw Exeption("Ошибка!", p, n); //Генерация исключения с значением исключения в виде объекта класса

}

}

this->p = p;

this->n = n;

for(int i = 0; i < n; ++i)

{

this->p[i] = p[i];

}

cout << "Данная последовательность является последовательностью Фибоначчи: ";

print();

cout << endl << endl;

}

void print() //Функция для печати всех данных

{

for(int i = 0; i < n; ++i)

{

cout << p[i] << " ";

}

}

~Fibonacci(){} //Деструктор

};

int main()

{

setlocale(LC\_ALL, "Rus"); //Реализация возможности печати русских букв

int arr1[6] = {0, 1, 1, 2, 3, 5}; //Массив с последовательностью Фибоначчи

int arr2[6] = { 0, 1, 1, 2, 4, 5 }; //Массив не являющийся последовательностью Фибоначчи

try //Контролируемый блок, в котором создаются объекты классов, содержащие только последовательности Фибоначчи

{

Fibonacci arr\_fib1(arr1, 6);

Fibonacci arr\_fib2(arr2, 6);

}

catch(Exeption &exep) //Обработка исключений

{

cout << exep.what() << endl; //Применение метода what() класса invalid\_argument

cout << "Данная последовательность чисел не является последовательностью Фибоначчи: ";

exep.print();

}

return 0;

}

Вывод:

Я научился использовать исключения в классе и обрабатывать их.