

Министерство образования Республики Беларусь

Учреждение образования

“Брестский государственный университет”

Кафедра ИИТ

Лабораторная работа №3

По дисциплине “Языки программирования”

Вариант №7

Выполнил:

Кравцевич Г.А. (ПО-7,1)

Проверил:

Дряпко. А. ?.

Дата выполнения:

21.09.21

Брест 2021

Цель:

Изучение правил перегрузки операций и принципов обработки исключений в C++.

Задание:

Написать программу, в которой описана иерархия классов: функция от одной переменной (синус, косинус, тангенс). Описать класс для хранения коллекции функций (массива указателей на базовый класс), в котором перегрузить операцию «[]». Описать класс- итератор для итерации по элементам коллекции. Для базового класса и его потомков перегрузить операции «==», «!=», «=». Продемонстрировать работу операторов.

Код программы:**Файл Function.h:**

```
#pragma once
#include <math.h>
#include <string>

using namespace std;

class Function
{
public:
    float value;
    float x;
    string name = "Simple function";

    Function();
    Function(float x);
    ~Function() { }

    virtual void setFuncValue(float x);

    string toString();
```

```

        void operator = (float v);
        bool operator == (Function rightFunction);
        bool operator != (Function rightFunction);

        Function operator / (Function rightFunction);
};

```

Файл Function.cpp:

```

#include "Function.h"

```

```

Function::Function()
{
    this->x = 0;
    this->value = 0;
}

```

```

Function::Function(float x)
{
    this->setFuncValue(x);
}

```

```

void Function::setFuncValue(float x)
{
    this->x = x;
    this->value = x;
}

```

```

string Function::toString()
{
    return this->name + "(" + to_string(this->x) + ") = " +
to_string(this->value);
}

```

```

bool Function::operator == (Function rightFunction)
{
    return this->value == rightFunction.value;
}

```

```

}

bool Function::operator != (Function rightFunction)
{
    return this->value != rightFunction.value;
}

void Function::operator = (float v)
{
    this->setFuncValue(v);
}

Function Function::operator / (Function rightFunction)
{
    Function result;
    result.value = this->value / rightFunction.value;

    return result;
}

```

Файл Cos.h:

```

#pragma once
#include "Function.h"

class Cos :
    virtual public Function
{
public:
    Cos();
    Cos(float x);

    void setFuncValue(float x);
};

```

Файл Cos.cpp:

```
#include "Cos.h"
```

```
Cos::Cos()
```

```
{  
    this->name = "Cos";  
    this->x = 0;  
    this->value = 0;  
}
```

```
Cos::Cos(float x) : Cos::Cos()
```

```
{  
    this->setFuncValue(x);  
}
```

```
void Cos::setFuncValue(float x)
```

```
{  
    this->x = x;  
    this->value = cos(x);  
}
```

Файл Sin.h:

```
#pragma once
```

```
#include "Function.h"
```

```
class Sin :
```

```
    virtual public Function
```

```
{
```

```
public:
```

```
    Sin();
```

```
    Sin(float x);
```

```
    void setFuncValue(float x);
```

```
};
```

Файл Sin.cpp:

```
#include "Sin.h"
```

```
Sin::Sin()
```

```
{  
    this->name = "Sin";  
    this->x = 0;  
    this->value = 0;  
}
```

```
Sin::Sin(float x) : Sin::Sin()
```

```
{  
    this->setFuncValue(x);  
}
```

```
void Sin::setFuncValue(float x)
```

```
{  
    this->x = x;  
    this->value = sin(x);  
}
```

Файл Tan.h:

```
#pragma once
```

```
#include "Function.h"
```

```
class Tan:
```

```
    virtual public Function
```

```
{
```

```
public:
```

```
    Tan();
```

```
    Tan(float x);
```

```
    void setFuncValue(float x);
```

```
};
```

Файл Tan.cpp:

```
#include "Tan.h"
```

```
Tan::Tan()
```

```
{  
    this->name = "Tan";  
    this->x = 0;  
    this->value = 0;  
}
```

```
Tan::Tan(float x) : Tan::Tan()
```

```
{  
    this->setFuncValue(x);  
}
```

```
void Tan::setFuncValue(float x)
```

```
{  
    this->x = x;  
    this->value = tan(x);  
}
```

Файл FuncCollection.h:

```
#pragma once
```

```
#include <iostream>
```

```
#include <vector>
```

```
#include "Function.h"
```

```
class FuncCollection
```

```
{  
public:  
    std::vector<Function> functions;
```

```

    void push(Function f);
    void remove(int index);

    void display();
    int length();

    Function operator [] (int index);
};

```

Файл FuncCollection.cpp:

```

#include "FuncCollection.h"

void FuncCollection::push(Function f)
{
    this->functions.push_back(f);
}

int FuncCollection::length()
{
    return this->functions.size();
}

void FuncCollection::remove(int index)
{
    if (index >= this->functions.size())
    {
        throw out_of_range("Index out of range exception");
    }

    this->functions.erase(this->functions.begin() + index);
}

Function FuncCollection::operator [] (int index)
{

```



```

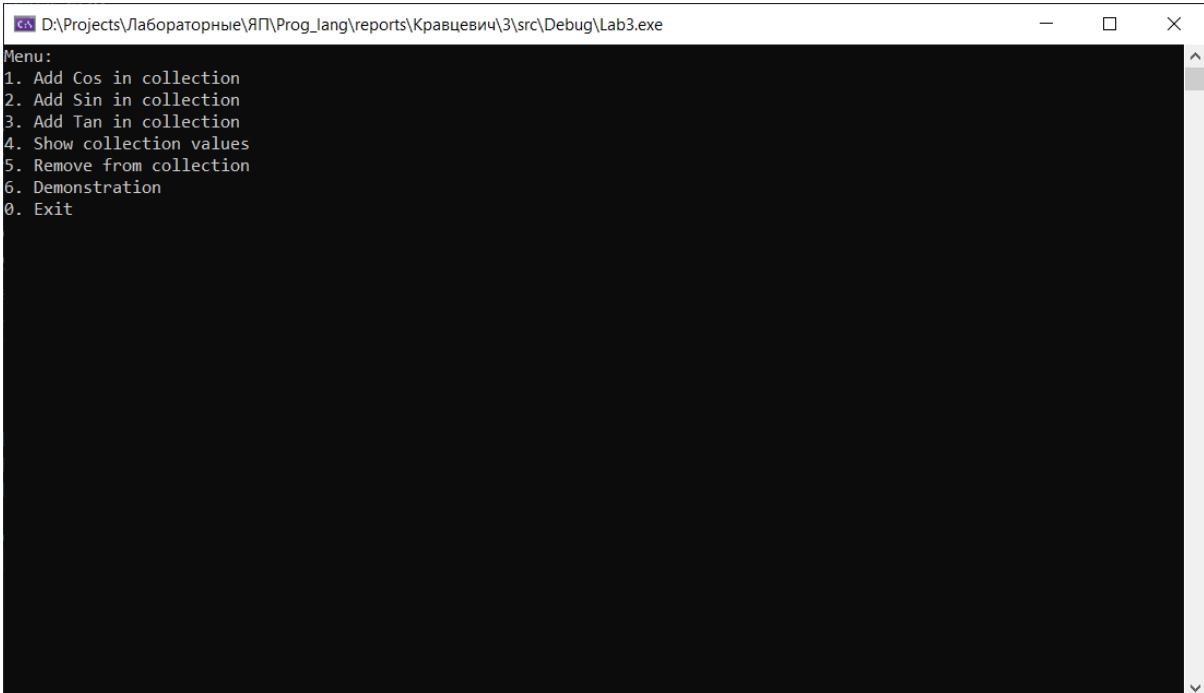
        if (index >= this->functions.size())
        {
            throw out_of_range("Index out of range exception");
        }

        return this->functions[index];
    }

void FuncCollection::display()
{
    for (size_t i = 0; i < this->length(); i++)
    {
        std::cout << i << ": " << this->functions[i].toString() << std::endl;
    }
}

```

Результат работы:



```

D:\Projects\Лабораторные\ЯП\Prog_lang\reports\Кравцевич\3\src\Debug\Lab3.exe
Menu:
1. Add Cos in collection
2. Add Sin in collection
3. Add Tan in collection
4. Show collection values
5. Remove from collection
6. Demonstration
0. Exit

```

```
D:\Projects\Лабораторные\ЯП\Prog_lang\reports\Кравцевич\3\src\Debug\Lab3.exe
Menu:
1. Add Cos in collection
2. Add Sin in collection
3. Add Tan in collection
4. Show collection values
5. Remove from collection
6. Demonstration
0. Exit
6
Values:
-0.416147
0.909297
-2.18504
cos == sin: 0
tg = sin/cos: 1
Для продолжения нажмите любую клавишу . . .
```