Министерство науки и высшего образования российской федерации федеральное государственное автономное образовательное учреждение высшего образования

«Северо-кавказский федеральный университет»

**Кафедра прикладной информатики**

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

ДИСЦИПЛИНЫ «Объектно-ориентированное программирование»

НА ТЕМУ:

Наследование

**Выполнила:**

студентка группы ПИН-б-о-21-1

ФИО Глущенко Марина Анатольевна

Проверил: Щеголев Алексей Алексеевич

Ставрополь, 2023

Задание: Написать программу, демонстрирующую работу с объектами двух типов, SymbString и OctString, для чего создать систему соответствующих классов. Каждый объект должен иметь идентификатор (в виде произвольной строки символов) и одно или несколько полей для хранения состояния (текущего значения) объекта. Клиенту (функции main) должны быть доступны следующие основные операции (методы): создать объект, удалить объект, показать значение объекта и прочие дополнительные операции (зависят от варианта). Операции по созданию и удалению объектов инкапсулировать в классе Factory. Предусмотреть меню, позволяющее продемонстрировать заданные операции.

На рисунке и листинге 1 продемонстрировано решение данной задачи.

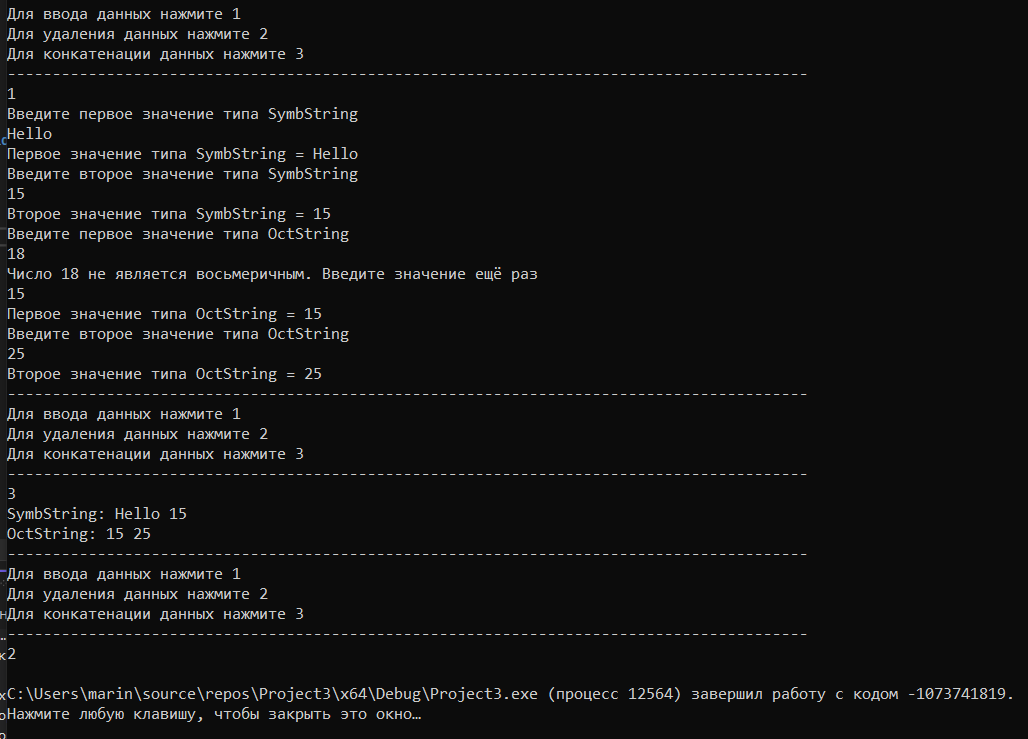


Рисунок 1 – Результат программы

Листинг 1 – Код программы

#include <iostream>

using namespace std;

#include <windows.h>

#include <string>

#include <cctype>

class Type

{

public:

virtual void Create1() = 0;

virtual void Create2() = 0;

virtual void Delete1() = 0;

virtual void Delete2() = 0;

virtual void Init() = 0;

};

class SymbString : public Type {

public:

string a = "";

string b = "";

void Create1() override {

cout << "Введите первое значение типа SymbString" << endl;

cin >> a;

cout << "Первое значение типа SymbString = " << a << endl;

}

void Create2() override {

cout << "Введите второе значение типа SymbString" << endl;

cin >> b;

cout << "Второе значение типа SymbString = " << b << endl;

}

void Init() override {

string s = a + " " + b;

cout << "SymbString: " << s << endl;

}

void Delete1() override {

delete this;

}

void Delete2() override {

delete this;

}

};

class OctString : public Type {

public:

string c;

string d;

void Create1() override {

cout << "Введите первое значение типа OctString" << endl;

cin >> c;

bool isOctal = true;

for (char digit : c) {

if (digit < '0' || digit > '7') {

isOctal = false;

break;

}

}

if (isOctal) {

cout << "Первое значение типа OctString = " << c << endl;

}

else {

std::cout << "Число " << c << " не является восьмеричным. Введите значение ещё раз" << std::endl;

cin >> c;

cout << "Первое значение типа OctString = " << c << endl;

}

cin.get();

}

void Create2() override {

cout << "Введите второе значение типа OctString" << endl;

cin >> d;

bool isOctal = true;

for (char digit : d) {

if (digit < '0' || digit > '7') {

isOctal = false;

break;

}

}

if (isOctal) {

cout << "Второе значение типа OctString = " << d << endl;

}

else {

std::cout << "Число " << d << " не является восьмеричным. Введите значение ещё раз" << std::endl;

cin >> d;

cout << "Второе значение типа OctString = " << d << endl;

}

cin.get();

}

void Init() override {

string m = c + " " + d;

cout << "OctString: " << m << endl;

}

void Delete1() override {

delete this;

}

void Delete2() override {

delete this;

}

};

class Look {

public:

virtual Type\* create() = 0;

};

class TypeSymbString : public Look{

public:

Type\* create() override {

return new SymbString();

}

};

class TypeOctString : public Look {

public:

Type\* create() override {

return new OctString();

}

};

void printHelp() {

printf("-----------------------------------------------------------------------------------------\n");

printf("Для ввода данных нажмите 1\n");

printf("Для удаления данных нажмите 2\n");

printf("Для конкатенации данных нажмите 3\n");

printf("-----------------------------------------------------------------------------------------\n");

}

int main(int argc, char\*\* argv)

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice = 0;

bool exit = false;

Look\* creator = new TypeSymbString();

Type\* SymbString = creator->create();

creator = new TypeOctString();

Type\* OctString = creator->create();

while (!exit) {

printHelp();

cin >> choice;

if (choice == 1) {

SymbString->Create1();

SymbString->Create2();

OctString->Create1();

OctString->Create2();

}

if (choice == 2) {

SymbString->Delete1();

SymbString->Delete2();

OctString->Delete1();

OctString->Delete2();

}

if (choice == 3) {

SymbString->Init();

OctString->Init();

}

}

return 0;

}