

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт  информационных технологий** | **Кафедра информационных систем** |

|  |  |  |
| --- | --- | --- |
| **Основная образовательная программа 09.03.02 «Информационные системы и технологии»**  **Отчет по дисциплине «Объектно-ориентированное программирование на языке высокого уровня»** | | |
| **по лабораторной работе №5** | | |
|  | | |
|  | | |
| **Проверил**  **ассистент** |  | **Михайлуца Н.А.** |
|  |  |  |
| **Выполнил**  **студент группы ИДБ-22-06** |  | **Мустафаева П.М.** |

ОГЛАВЛЕНИЕ

[ЗАДАНИЕ 1 2](#_Toc149855007)

[1.1 ОПИСАНИЕ 2](#_Toc149855008)

[1.2 ПРОГРАММА 3](#_Toc149855009)

[1.3 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ 4](#_Toc149855010)

[ЗАДАНИЕ 2 4](#_Toc149855011)

[2.1 ОПИСАНИЕ 4](#_Toc149855012)

[2.2 ПРОГРАММА 5](#_Toc149855013)

[2.3 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ 6](#_Toc149855014)

[ЗАКЛЮЧЕНИЕ 8](#_Toc149855015)

# **ЗАДАНИЕ 1**

## **1.1 ОПИСАНИЕ**

Задать полиморфную иерархию из 2х классов. Кроме того, задать глобальный массив g\_Array (не STL-контейнер, простой массив) (динамический — это важно!) для хранения 5 элементов объектов заданных типов. В главной функции создать в циклах 10 объектов заданных типов, помещая их посредством специальной функции append в g\_Array (подсказка: в определенный момент надо будет выделить достаточную память для нового хранилища g\_Array и скопировать туда данные из старой памяти ).

Далее в главной функции продемонстрировать различное поведение объектов, собранных в g\_Array, и затем освободить ресурсы на выходе из программы.

## **1.2 ПРОГРАММА**

#include <iostream>

using namespace std;

class Base

{

public:

bool\* isBase = new bool;

Base() {

\*isBase = true;

}

Base(const Base& obj) {

\*isBase = \*obj.isBase;

}

virtual Base\* copy() {

return new Base(\*this);

}

void print() {

if (\*isBase) {

cout << "Base\n";

}

else {

cout << "Derived\n";

}

}

virtual ~Base() {

delete isBase;

};

};

class Derived : public Base {

public:

Derived() {

\*isBase = false;

}

Derived\* copy() {

return new Derived(\*this);

}

Derived(const Derived& obj) : Base(obj) {

\*isBase = \*obj.isBase;

}

~Derived() {};

};

Base\*\* g\_Array = new Base \* [5];

void append(Base\* obj, int i) {

g\_Array[i] = obj;

}

int main()

{

int size = 5;

for (int i = 0; i < size; i++) {

if (i % 2 == 0) {

append(new Base, i);

}

else {

append(new Derived, i);

}

}

Base\*\* new\_array = new Base \* [10];

for (int i = 0; i < 5; i++)

new\_array[i] = g\_Array[i]->copy();

for (int i = 5; i < 10; i++) {

new\_array[i] = g\_Array[i - 5]->copy();

delete g\_Array[i - 5];

}

delete[] g\_Array;

g\_Array = new\_array;

for (int i = 0; i < 10; i++) {

g\_Array[i]->print();

}

for (int i = 0; i < 10; i++) {

delete g\_Array[i];

}

delete[] g\_Array;

}

## **1.3 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ**

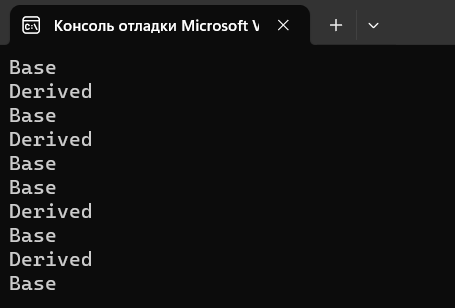


Рис. 1. Результат выполнения программы 1

# **ЗАДАНИЕ 2**

## **2.1 ОПИСАНИЕ**

Реализовать односвязный список объектов обобщенного типа. В функции main продемонстрировать его функциональные возможности, заполняя список, выводя данные на консоль, а также корректно освобождая ресурсы.

## **2.2 ПРОГРАММА**

#include <iostream>

using namespace std;

template <typename T>

class Node {

public:

T data;

Node<T>\* next;

Node(T data) {

this->data = data;

this->next = nullptr;

}

};

template <typename T>

class LinkedList {

private:

Node<T>\* head;

public:

LinkedList() {

this->head = nullptr;

}

void add(T data) {

Node<T>\* newNode = new Node<T>(data);

if (head == nullptr) {

head = newNode;

}

else {

Node<T>\* current = head;

while (current->next != nullptr) {

current = current->next;

}

current->next = newNode;

}

}

void display() {

Node<T>\* current = head;

while (current != nullptr) {

cout << current->data << " ";

current = current->next;

}

cout << endl;

}

void clear() {

Node<T>\* current = head;

while (current != nullptr) {

Node<T>\* next = current->next;

delete current;

current = next;

}

head = nullptr;

}

};

int main() {

LinkedList<string> list;

list.add("One");

list.add("Two");

list.add("Three");

list.add("Four");

list.display();

list.clear();

return 0;

}

## **2.3 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ**

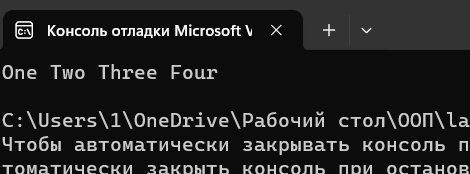


Рис. 2. Результат выполнения программы 2

# **ЗАКЛЮЧЕНИЕ**

В ходе лабораторной работы была реализована работа с односвязными списками обобщенного вида, массивами, а так же продемонстрировали функциональные возможности программ.