**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И.Раззакова**

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра: **Программное обеспечение компьютерных систем**

Курс «Объектно-ориентированное программирование»

**ОТЧЕТ**

**Лабораторная работа №7**

Выполнил: студент группы ПИ-3-21

Урманбетов Султан

Проверил: Мусабаев Э.Б.

**Бишкек 2024**

# **Задание №1**

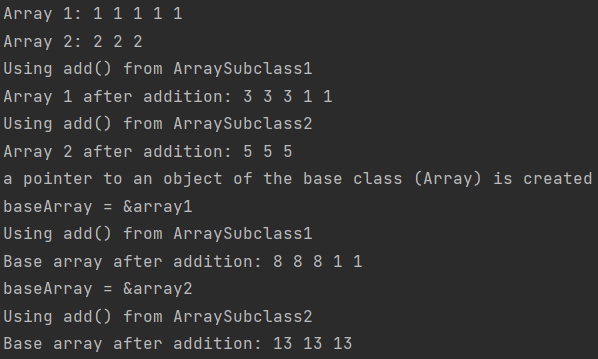
**Постановка задачи:**

Создать базовый класс Array с полями: массив типа unsigned и поле для хранения количества элементов у текущего объекта массива. Максимально возможный размер массива задается статической константой. Реализовать конструктор инициализации, задающий количество элементов и начальное значение (по умолчанию 0). Реализовать в классе Array виртуальную функцию поэлементного сложения массивов. Реализовать два класса, переопределив виртуальную функцию сложения. Вызывающая программа должна продемонстрировать все варианты вызова виртуальных функций.

**Исходный код программы на C++:**

#include <iostream>  
#include <fstream>  
  
class Array {  
protected:  
 static const int MAX\_SIZE = 100; // Максимальный размер массива  
 unsigned int elements[MAX\_SIZE]; // Массив элементов  
 unsigned int count; // Количество элементов в массиве  
  
public:  
 // Конструктор инициализации  
 Array(unsigned int size, unsigned int init\_value = 0) : count(size) {  
 for (int i = 0; i < size; ++i) {  
 elements[i] = init\_value;  
 }  
 }  
  
 // Виртуальная функция поэлементного сложения массивов  
 virtual void add(const Array& other) {  
 unsigned int min\_count = std::min(count, other.count);  
 for (int i = 0; i < min\_count; ++i) {  
 elements[i] += other.elements[i];  
 }  
 }  
  
 // Вывод элементов массива  
 void display() const {  
 for (int i = 0; i < count; ++i) {  
 std::cout << elements[i] << " ";  
 }  
 std::cout << std::endl;  
 }  
  
 // Заполнение массива данными из файла  
 void fillFromFile(std::ifstream& file) {  
 for (int i = 0; i < count; ++i) {  
 file >> elements[i];  
 }  
 }  
};  
  
// Первый подкласс  
class ArraySubclass1 : public Array {  
public:  
 ArraySubclass1(unsigned int size, unsigned int init\_value = 0) : Array(size, init\_value) {}  
  
 // Переопределение виртуальной функции сложения  
 void add(const Array& other) override {  
 Array::add(other); // Вызов базовой реализации  
 std::cout << "Using add() from ArraySubclass1" << std::endl;  
 }  
};  
  
// Второй подкласс  
class ArraySubclass2 : public Array {  
public:  
 ArraySubclass2(unsigned int size, unsigned int init\_value = 0) : Array(size, init\_value) {}  
  
 // Переопределение виртуальной функции сложения  
 void add(const Array& other) override {  
 Array::add(other); // Вызов базовой реализации  
 std::cout << "Using add() from ArraySubclass2" << std::endl;  
 }  
};  
  
int main() {  
 std::ifstream inputFile("input.txt"); // Открытие файла для чтения  
  
 unsigned int size1, size2;  
 if (inputFile.is\_open()) {  
 inputFile >> size1 >> size2; // Чтение размеров массивов из файла  
  
 ArraySubclass1 array1(size1, 1); // Создание объекта первого подкласса  
 ArraySubclass2 array2(size2, 2); // Создание объекта второго подкласса  
  
 // Заполнение массивов данными из файла  
 array1.fillFromFile(inputFile);  
 array2.fillFromFile(inputFile);  
  
 inputFile.close(); // Закрытие файла  
  
  
 std::cout << "Array 1: ";  
 array1.display();  
 std::cout << "Array 2: ";  
 array2.display();  
  
 // Вызов виртуальной функции поэлементного сложения для array1  
 array1.add(array2);  
 std::cout << "Array 1 after addition: ";  
 array1.display();  
  
 // Вызов виртуальной функции поэлементного сложения для array2  
 array2.add(array1);  
 std::cout << "Array 2 after addition: ";  
 array2.display();  
  
 // Создание объекта базового класса и вызов виртуальной функции  
 std::cout << "a pointer to an object of the base class (Array) is created"<<std::endl;  
 Array\* baseArray;  
 std::cout << "baseArray = &array1"<<std::endl;  
 baseArray = &array1;  
 baseArray->add(array2);  
 std::cout << "Base array after addition: ";  
 baseArray->display();  
  
 baseArray = &array2;  
 std::cout << "baseArray = &array2"<<std::endl;  
 baseArray->add(array1);  
 std::cout << "Base array after addition: ";  
 baseArray->display();  
 } else {  
 std::cerr << "Unable to open input file." << std::endl;  
 return 1;  
 }  
  
  
 return 0;  
}

**Тест:**

****

# **Задание №1**

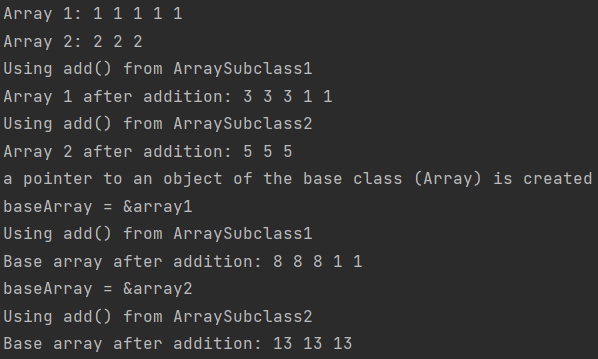
**Постановка задачи:**

Создать базовый класс Array с полями: массив типа unsigned и поле для хранения количества элементов у текущего объекта массива. Максимально возможный размер массива задается статической константой. Реализовать конструктор инициализации, задающий количество элементов и начальное значение (по умолчанию 0). Реализовать в классе Array виртуальную функцию поэлементного сложения массивов. Реализовать два класса, переопределив виртуальную функцию сложения. Вызывающая программа должна продемонстрировать все варианты вызова виртуальных функций.

**Исходный код программы на C++:**

#include <iostream>  
#include <fstream>  
  
class Array {  
protected:  
 static const int MAX\_SIZE = 100; // Максимальный размер массива  
 unsigned int elements[MAX\_SIZE]; // Массив элементов  
 unsigned int count; // Количество элементов в массиве  
  
public:  
 // Конструктор инициализации  
 Array(unsigned int size, unsigned int init\_value = 0) : count(size) {  
 for (int i = 0; i < size; ++i) {  
 elements[i] = init\_value;  
 }  
 }  
  
 // Виртуальная функция поэлементного сложения массивов  
 virtual void add(const Array& other) {  
 unsigned int min\_count = std::min(count, other.count);  
 for (int i = 0; i < min\_count; ++i) {  
 elements[i] += other.elements[i];  
 }  
 }  
  
 // Вывод элементов массива  
 void display() const {  
 for (int i = 0; i < count; ++i) {  
 std::cout << elements[i] << " ";  
 }  
 std::cout << std::endl;  
 }  
  
 // Заполнение массива данными из файла  
 void fillFromFile(std::ifstream& file) {  
 for (int i = 0; i < count; ++i) {  
 file >> elements[i];  
 }  
 }  
};  
  
// Первый подкласс  
class ArraySubclass1 : public Array {  
public:  
 ArraySubclass1(unsigned int size, unsigned int init\_value = 0) : Array(size, init\_value) {}  
  
 // Переопределение виртуальной функции сложения  
 void add(const Array& other) override {  
 Array::add(other); // Вызов базовой реализации  
 std::cout << "Using add() from ArraySubclass1" << std::endl;  
 }  
};  
  
// Второй подкласс  
class ArraySubclass2 : public Array {  
public:  
 ArraySubclass2(unsigned int size, unsigned int init\_value = 0) : Array(size, init\_value) {}  
  
 // Переопределение виртуальной функции сложения  
 void add(const Array& other) override {  
 Array::add(other); // Вызов базовой реализации  
 std::cout << "Using add() from ArraySubclass2" << std::endl;  
 }  
};  
  
int main() {  
 std::ifstream inputFile("input.txt"); // Открытие файла для чтения  
  
 unsigned int size1, size2;  
 if (inputFile.is\_open()) {  
 inputFile >> size1 >> size2; // Чтение размеров массивов из файла  
  
 ArraySubclass1 array1(size1, 1); // Создание объекта первого подкласса  
 ArraySubclass2 array2(size2, 2); // Создание объекта второго подкласса  
  
 // Заполнение массивов данными из файла  
 array1.fillFromFile(inputFile);  
 array2.fillFromFile(inputFile);  
  
 inputFile.close(); // Закрытие файла  
  
  
 std::cout << "Array 1: ";  
 array1.display();  
 std::cout << "Array 2: ";  
 array2.display();  
  
 // Вызов виртуальной функции поэлементного сложения для array1  
 array1.add(array2);  
 std::cout << "Array 1 after addition: ";  
 array1.display();  
  
 // Вызов виртуальной функции поэлементного сложения для array2  
 array2.add(array1);  
 std::cout << "Array 2 after addition: ";  
 array2.display();  
  
 // Создание объекта базового класса и вызов виртуальной функции  
 std::cout << "a pointer to an object of the base class (Array) is created"<<std::endl;  
 Array\* baseArray;  
 std::cout << "baseArray = &array1"<<std::endl;  
 baseArray = &array1;  
 baseArray->add(array2);  
 std::cout << "Base array after addition: ";  
 baseArray->display();  
  
 baseArray = &array2;  
 std::cout << "baseArray = &array2"<<std::endl;  
 baseArray->add(array1);  
 std::cout << "Base array after addition: ";  
 baseArray->display();  
 } else {  
 std::cerr << "Unable to open input file." << std::endl;  
 return 1;  
 }  
  
  
 return 0;  
}

**Тест:**

****

# **Ответы на вопросы:**