**Министерство науки и высшего образования Российской Федерации**Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича Столетовых»**

**(ВлГУ)**

Институт прикладной математики, физики и информатики

Кафедра физики и прикладной математики

**ЛАБОРАТОРНАЯ РАБОТА №5**

по дисциплине  
«Основы программирования»  
на тему:  
**«Указатели и ссылки. Одномерные массивы»**

Выполнил:

ст. гр. ПМИ-123

Рушев А.М.

Принял:  
ст. преподаватель

каф. ФиПМ

Черников А.С.

Владимир, 2023 г.

**Цель работы:** освоить принципы работы с указателями и ссылками в языке С++. Изучить операции, выполнение которых допустимые на указателях и ссылках. Получить навыки работы с массивами, в том числе, организации циклического перебора элементов, приёмов накопления.

**Задание:**

1) Объявить и проинициализировать переменные следующих типов: int, float, char. Объявить переменную типа void. Объявить указатели и ссылки на все перечисленные выше типы, настроить указатели и ссылки на объявленные переменные базовых типов. Вывести значения переменных на экран, обращаясь к ним через указатели.

2) Выполнить перенастройку указателей на переменные других типов, используя явное и неявное приведение типов указателей.

3) Привести примеры допустимых операций над указателями и ссылками. Пояснить смысл данных операций.

4) Объявить и проинициализировать следующие переменные и константы:

* константный указатель на int;
* указатель на целочисленную константу;
* указатель на указатель на int;
* константный указатель на указатель на int;
* указатель на константный указатель на целое;
* указатель на указатель на целочисленную константу;
* константный указатель на указатель на целочисленную константу;
* константный указатель на константный указатель на int;
* константный указатель на константный указатель на целочисленную константу;

Пояснить схематично организованные цепочки данных.

5) Объявить и проинициализировать одномерный целочисленный массив. Вычислить и вывести на экран сумму элементов этого массива, больших нуля.

6) Объявить и проинициализировать вводом с клавиатуры одномерный массив вещественных чисел. Вычислить и вывести на экран разность квадратов максимального и минимального значения элементов этого массива.

7) Объявить одномерный целочисленный массив. Вычислить среднее арифметическое элементов, обращаясь к ним через имя массива, используя его как указатель. Объявить два указателя на целое, настроить один из них на максимальный элемент массива, другой на минимальный. Вывести значения максимального и минимального элемента на экран, обращаясь к ним через эти указатели.

**Практическая часть:**

**Задание 1.**

Листинг программы:

void task1() {

cout << "\nЗадание 1." << endl;

int a = 13;

float b = 9.5;

char ch = 'C';

void\* v = nullptr;

int\* Aa = &a;

float\* Bb = &b;

char\* Ch = &ch;

int& Aa1 = a;

float& Bb1 = b;

char& Ch1 = ch;

cout << "Значение int, полученное через указатель: " << \*Aa << endl;

cout << "Значение float, полученное через указатель: " << \*Bb << endl;

cout << "Значение char, полученное через указатель: " << \*Ch << endl;

}

Результат выполнения кода:

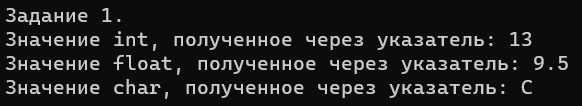


Рисунок 1 – Результат выполнения кода задания 1

**Задание 2.**

Листинг программы:

void task2() {

cout << "\nЗадание 2." << endl;

int a = 13;

float b = 9.5;

char ch = 'C';

int\* Aa = &a;

float\* Bb = reinterpret\_cast<float\*>(&a);

char\* Ch = reinterpret\_cast<char\*>(&a);

cout << "Значение integer, полученное через указатель: " << \*Aa << endl;

cout << "Значение integer, полученное через указатель Bb (явное приведение): " << \*Bb << endl;

cout << "Значение integer, полученное через указатель Ch (явное приведение): " << \*Ch << endl;

\*Bb = 9.5;

\*Ch = 'C';

cout << "Новое значение integer после явного приведения: " << a << endl;

Bb = reinterpret\_cast<float\*>(&Bb);

Ch = reinterpret\_cast<char\*>(&Bb);

cout << "Значение floating, полученное через указатель Bb: " << \*Bb << endl;

cout << "Значение floating, полученное через указатель Ch (неявное приведение): " << \*Ch << endl;

}

Результат выполнения кода:

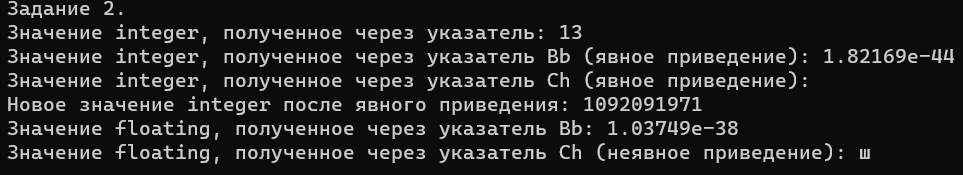


Рисунок 2 – Результат выполнения кода задания 2

**Задание 3.**

Листинг программы:

void task3() {

cout << "\nЗадание 3." << endl;

int value = 10;

int\* a = &value; // объявление и инициализация указателя на переменную value

int& b = value; // объявление и инициализация ссылки на переменную value

// Вывод значения переменной через указатель

cout << "Значение переменной value через указатель: " << \*a << std::endl;

// Изменение значения переменной через указатель

\*a = 20;

cout << "Новое значение переменной value: " << value << std::endl;

// Вывод значения переменной через ссылку

cout << "Значение переменной value через ссылку: " << b << std::endl;

// Изменение значения переменной через ссылку

b = 30;

cout << "Новое значение переменной value: " << value << std::endl;

}

Результат выполнения кода:

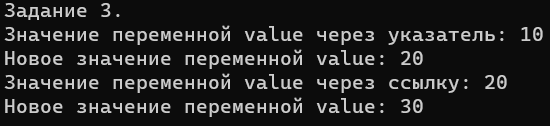


Рисунок 3 – Результат выполнения кода задания 3

**Задание 4.**

Листинг программы:

void task4() {

cout << "\nЗадание 4." << endl;

int num = 5;// переменная int

int\* ptr1 = &num;// константный указатель на int

const int\* ptr2 = &num;// указатель на целочисленную константу

int\*\* ptr3 = &ptr1;// указатель на указатель на int

int\*\* const ptr4 = &ptr1;// константный указатель на указатель на int

const int\*\* ptr5 = &ptr2;// указатель на константный указатель на целое

const int\*\* const ptr6 = &ptr2;// указатель на указатель на целочисленную константу

const int\*\* const ptr7 = &ptr2;// константный указатель на указатель на целочисленную константу

const int\* const\* ptr8 = &ptr2;// константный указатель на константный указатель на int

const int\* const\* const ptr9 = &ptr2;// константный указатель на константный указатель на целочисленную константу

cout << "num: " << num << endl;

cout << "\*ptr1: " << \*ptr1 << endl;

cout << "\*ptr2: " << \*ptr2 << endl;

cout << "\*\*ptr3: " << \*\*ptr3 << endl;

cout << "\*\*ptr4: " << \*\*ptr4 << endl;

cout << "\*ptr5: " << \*\*ptr5 << endl;

cout << "\*\*ptr6: " << \*\*ptr6 << endl;

cout << "\*\*ptr7: " << \*\*ptr7 << endl;

cout << "\*ptr8: " << \*\*ptr8 << endl;

cout << "\*\*ptr9: " << \*\*ptr9 << endl;

}

Результат выполнения кода:

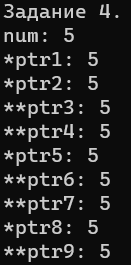


Рисунок 4 – Результат выполнения кода задания 4

* num — это переменная типа int.
* ptr1 — это константный указатель на int. Он указывает на адрес переменной num, и может использоваться для доступа к ее значению и изменения его значения.
* ptr2 — это указатель на целочисленную константу. Он указывает на адрес переменной num и должен использоваться только для доступа к значению переменной num. Нельзя использовать этот указатель для изменения значения переменной num.
* ptr3 — это указатель на указатель на int. Он указывает на адрес ptr1, который, в свою очередь, указывает на адрес переменной num. Используя ptr3 и двойную разименованность (\*\*ptr3), мы можем получить доступ к значению переменной num.
* ptr4 — это константный указатель на указатель на int. Он указывает на адрес ptr1 и имеет const-квалификатор, что означает, что он не может быть использован для изменения адреса, на который он указывает, т.е. адреса переменной num. Пример продолжается аналогично, объявляя указатели на указатели и константы указателей. Мы также можем использовать эти указатели для доступа к значению переменной num, используя различные уровни разименования. При выполнении кода выше мы увидим, что все указатели правильно настроены и позволяют доступ к значению переменной num.

1. int num
2. const int \*ptr1 -> &num
3. const int \*ptr2 -> &num
4. int \*\*ptr3 -> &ptr1 -> &num
5. const int \*\*ptr4 -> &ptr1

**Задание 5.**

Листинг программы:

void task5() {

cout << "\nЗадание 5." << endl;

int sum = 0;

int mass[n] = { 1,-3,4,2,-12 };

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

cout << "mass[" << i << "]" << mass[i] << endl;

if (mass[i] > 0) {

sum += mass[i];

}

}

cout << "sum = " << sum << endl;

}

Результат выполнения кода:

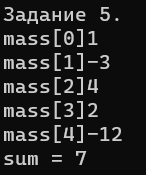


Рисунок 5 – Результат выполнения кода задания 5

**Задание 6.**

Листинг программы:

void task6() {

double mass[n];

double max = mass[n], min = mass[n], x;

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

cout << "Введите значение для " << i << " элемента массива: ";

cin >> mass[i];

}

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

if (mass[i] > max) {

max = mass[i];

}

if (mass[i] < min) {

min = mass[i];

}

}

cout << "Максимальный элемент массива: " << max << endl;

cout << "Минимальный элекмент массива: " << min << endl;

x = (max \* max) - (2 \* max \* min) + (min \* min);

cout << "Разность квадратов максимального и минимального элементов = " << x << endl;

}

Результат выполнения кода:

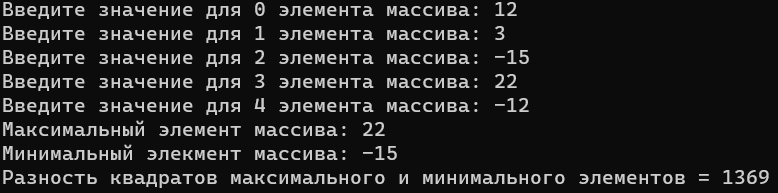


Рисунок 6 – Результат выполнения кода задания 6

**Задание 7.**

Листинг программы:

void task7() {

cout << "\nЗадание 7." << endl;

int mass[n];

//заполенение массива

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

mass[i] = 1 + rand() % 100;

cout << "mass: " << mass[i] << endl;

}

// Вычисление среднего арифметического

double sum = 0;

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

sum += mass[i];

}

double rage = sum / n;

cout << "Среднее арифметическое: " << rage << endl;

// Поиск максимального и минимального элементов

int\* maxMass = &mass[0];

int\* minMass = &mass[0];

for (int i = 1; i < n; i++) {

if (mass[i] > \*maxMass) {

maxMass = &mass[i];

}

if (mass[i] < \*minMass) {

minMass = &mass[i];

}

}

//\*(mas + i) > \*maxMass

// Вывод максимального и минимального элементов

std::cout << "Максимальный элемент: " << \*maxMass << std::endl;

std::cout << "Минимальный элемент: " << \*minMass << std::endl;

}

Результат выполнения кода:

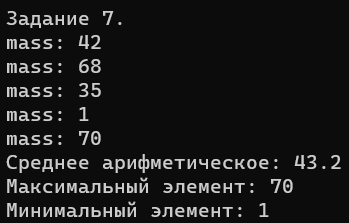


Рисунок 7 – Результат выполнения кода задания 7

**Вывод:** освоили принципы работы с указателями и ссылками в языке С++. Изучили операции, выполнения которых допустимы на указателях и ссылках. Получили навыки работы с массивами, в том числе, организации циклического перебора элементов, приёмов накопления.