Задача 1. Обработка одномерных массивов.

Цель работы: изучить и применить метод вставок для сортировки одномерного массива целых чисел по неубыванию количества цифр в элементах массива.

Выполнение: схема алгоритма изображена на рисунке 1. Текст программы – на рисунке 2.

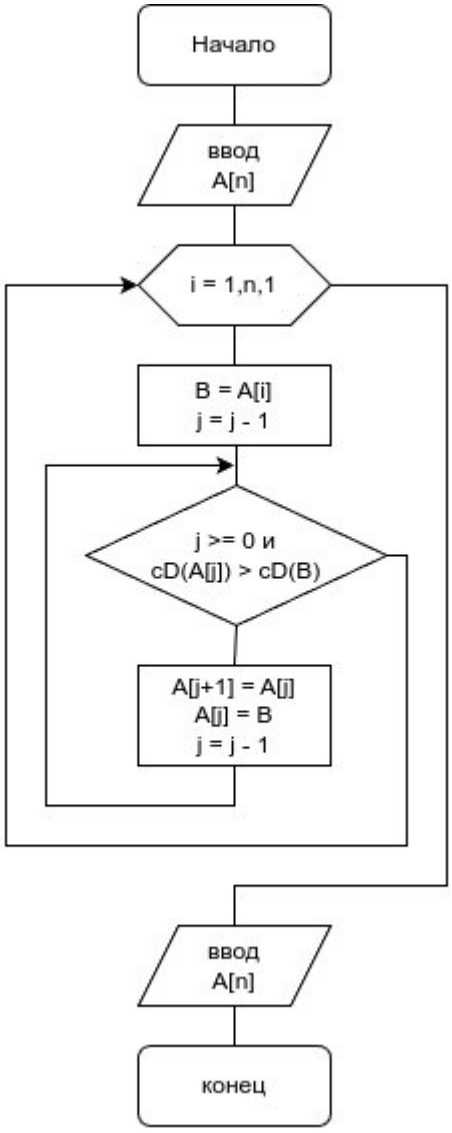

Рисунок 1 — схема алгоритма

```cpp
int countDigits(int num) {
    int result = 0;
    do {
        num /= 10;
        result++;
    } while (num != 0);
    return result;
}

int main() {
    const int N = 10;
    int nums[N];
    cout << "Enter nums (" << N << " elements):\n";
    for (int &num: nums)
        cin >> num;

    for (int i = 1; i < N; ++i) {
        int key = nums[i];
        int j = i - 1;
        while (j >= 0 && countDigits(nums[j]) > countDigits(key)) {
            nums[j + 1] = nums[j];
            nums[j] = key;
            j--;
        }
    }

    for (const auto &elem:const int& : nums) {
        cout << elem << " ";
    }
    return 0;
}
```

Рисунок 2 — код программы

Тестовые данные и результаты тестирования: тестовые данные и результаты тестирования изображены на рисунке 3

Ввод	Вывод
1 4 1 32 433 43 4333 12 1 222	1 4 1 1 32 43 12 433 222 4333
434 43 -12 3242 429434 12 1 23 30 2	1 2 43 -12 12 23 30 434 3242 429434

Рисунок 3 — Тестовые данные и результаты тестирования

Вывод: В ходе выполнения лабораторной работы была реализована и протестирована сортировка одномерного массива методом вставок. Задача позволила закрепить знания о методах сортировки и углубить понимание работы с массивами в языке программирования C++.

Задача 2. Обработка строк

Цель работы: Изучить методы обработки строк на языке C++ и разработать программный модуль для подсчета количества слов, содержащих более трех гласных букв, в заданной последовательности слов.

Выполнение: текст программы изображен на рисунке 4

```
int countVowels(const string &str) {
    int result = 0;
    for (const auto &c:const char& : str) {
        if (vowels.find(c) != string::npos)
            result++;
    }
    return result;
}

int main() {
    string text;
    int result = 0;

    cout << "Enter text: \n";
    getline([&]cin, [&]text);
    text += ' ';
    string word;

    for (const auto &c:const char& : text) {
        if (alphabet.find(c) != string::npos)
            word += c;
        else {
            if (!word.empty() && countVowels(word) > 3)
                result++;
            word = "";
        }
    }

    cout << "Words with 3+ vowels: " << result << endl;
    return 0;
}
```

Рисунок 4 — код программы

Тестовые данные и результаты тестирования: тестовые данные и результаты тестирования изображены на рисунке 5

Ввод	Вывод
a aa aAa AaaA aaaaA	Words with 3+ vowels: 2
Hello, beautiful world.	Words with 3+ vowels: 1

Рисунок 5 — тестовые данные и результат выполнения

Вывод: В ходе лабораторной работы была успешно реализована программа, которая обрабатывает строку, подсчитывает количество гласных в каждом слове и определяет количество слов, содержащих более трех гласных букв. Это позволило углубить понимание работы со строками и их обработки в языке программирования C++.

Задача 3. Создание псевдомодулей. Процедурный тип параметров

Цель работы: Разработать и реализовать подпрограмму, вычисляющую значение данного выражения с использованием функций, передаваемых через параметры, и применить её для вычисления конкретных значений выражения при заданных параметрах.

Выполнение: текст программы изображен на рисунках 6 и 7

```
double f1_1(double x) {
    return sin(x) * x;
}

double f2_1(double x) {
    return 0.64 * cos(x);
}

double f3_1(double x) {
    return pow(x, y:3) - 5;
}

double f1_2(double x) {
    return log(x) * x;
}

double f2_2(double x) {
    return cos(x) * pow(x, y:2);
}

double f3_2(double x) {
    return pow(x, y:4) + 4;
}



int main() {
    const double x1 = -3.5, x2 = 3;
    cout << "x1: " << x1 << ". Result: " << countP(x1, f1:f1_1, f2:f2_1, f3:f3_1) << endl;
    cout << "x2: " << x2 << ". Result: " << countP(x2, f1:f1_2, f2:f2_2, f3:f3_2) << endl;
    return 0;
}
```

Рисунок 6 — текст основной программы

```
double countP(double x, double (*f1)(double), double (*f2)(double), double (*f3)(double)) {
    return f1(x) * pow(x, y:2) + f2(x) * x + f3(x);
}
```

Рисунок 7 — текст программы модуля

Тестовые данные и результаты тестирования:

x1: -3.5. Result: -60.8172

x2: 3. Result: 87.9327

Вывод: В ходе работы была успешно разработана подпрограмма для вычисления значения выражения с параметрами, передаваемыми через функциональные аргументы. Написанная программа была протестирована на двух различных наборах параметров, что позволило закрепить навыки работы с функциональными аргументами и процедурным типом параметров в языке программирования C++.