МИНИСТЕРСТВО НАУКИ ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный университет геодезии и картографии»
(МИИГАиК)
Факультет геоинформатики и информационной безопасности
Кафедра геоинформационных систем и технологий

Лабораторная работа №4

"Разработка консольного приложения с использованием функций"

Проверил:
Лебедев Евгений Денисович

Выполнил:
Студент группы: 2024-ФГиИБ-ПИ-1б
Шамадаев Рустам Эльдарович

Москва 2024

Задание номер 30:

Добавить к программе реализацию данных задач:

Заданы два массива A(5) и B(5). Подсчитать в них количество положительных элементов и первым на печать вывести массив, имеющий наибольшее их количество.

Даны два множества A и B. Найти множество, которое состоит из элементов, присутствующих в обоих множествах и являющихся числами, у которых все цифры одинаковые.

Ссылка на Github: https://github.com/m1lle3r/inf/tree/main

Листинг 1:

Функция countPositive инициализирует переменную count значением 0 и в цикле перебирает все элементы массива. Каждый раз, когда встречается положительное число (большее нуля), счетчик увеличивается на 1. Результат записывается в count.

```
void countPositive(const int* arr, int size, int& count) {
    count = 0;
    for (int i = 0; i < size; i++) {
        if (arr[i] > 0) {
            count++;
        }
    }
}
```

Функция findArrayWithMaxPositive сначала вызывает countPositive для каждого массива, чтобы подсчитать количество положительных элементов. Затем она сравнивает эти количества. Если в первом массиве (arrA) больше положительных элементов, то выводится этот массив, иначе — второй массив (arrB).

```
void findArrayWithMaxPositive(const int* arrA, const int* arrB, int size) {
    int countA = 0, countB = 0;
```

```
    countPositive(arrA, size, countA);
    countPositive(arrB, size, countB);
```

```
    if (countA > countB) {
        std::cout << "Массив A имеет больше положительных элементов: ";
        for (int i = 0; i < size; i++) {
            std::cout << arrA[i] << " ";
        }
    } else {
        std::cout << "Массив B имеет больше положительных элементов: ";
        for (int i = 0; i < size; i++) {
            std::cout << arrB[i] << " ";
        }
    }
    std::cout << std::endl;
}
```

Массив A: {1, 2, 3, 4, 5}

Массив B: {-1, -2, -3, -4, -5}

Результат: Массив A имеет наибольшую сумму (15) Массив A имеет больше положительных элементов (5)

Массив A: {0, -1, -2, 3, 4}

Массив B: {5, -3, 2, 0, -1}

Результат: Массив A имеет наибольшую сумму (4) Массивы имеют одинаковое количество положительных элементов (2)

Массив A: {10, -3, 5, -7, 2}

Массив B: {1, -4, 6, 3, 5}

Результат: Массив B имеет наибольшую сумму (11) Массив B имеет больше положительных элементов (4)

Листинг 2:

Функция hasIdenticalDigits сначала преобразует число в строку с помощью std::to_string. Затем она проверяет, все ли цифры числа одинаковы. Для этого она сравнивает каждую цифру числа с первой цифрой. Если хотя бы одна цифра отличается от первой, функция возвращает false, иначе — true.

```
bool hasIdenticalDigits(int num) {
    std::string str = std::to_string(num);
    for (size_t i = 1; i < str.length(); i++) {
        if (str[i] != str[0]) {
            return false;
        }
    }
    return true;
}
```

Функция findCommonIdenticalDigitNumbers находит пересечение двух множеств с помощью цикла. Для каждого числа из пересечения вызывается функция hasIdenticalDigits. Если число состоит из одинаковых цифр, оно добавляется в результирующее множество. Пересечение производится с использованием метода count, который проверяет наличие элемента во втором множестве.

```
void findCommonIdenticalDigitNumbers(const std::set<int>& setA, const std::set<int>& setB) {
    std::set<int> intersection;

    for (const int& num : setA) {
        if (setB.count(num) > 0 && hasIdenticalDigits(num)) {
            intersection.insert(num);
        }
    }

    std::cout << "Пересечение с одинаковыми цифрами: ";
    for (const int& num : intersection) {
        std::cout << num << " ";
    }
    std::cout << std::endl;
}
```

Множество A: {111, 222, 333}
Множество B: {111, 444, 555}
Результат: Пересечение с одинаковыми цифрами: 111

Множество A: {1, 2, 3, 4, 5}
Множество B: {6, 7, 8, 9, 10}
Результат: Пересечение с одинаковыми цифрами: (пусто)

Множество A: {111, 123, 999}
Множество B: {111, 555, 999}
Результат: Пересечение с одинаковыми цифрами: 111 999

Результат: Массив B имеет наибольшую сумму (11) Массив B имеет больше положительных элементов (4)

Листинг 2:

Функция hasIdenticalDigits сначала преобразует число в строку с помощью std::to_string. Затем она проверяет, все ли цифры числа одинаковы. Для этого она сравнивает каждую цифру числа с первой цифрой. Если хотя бы одна цифра отличается от первой, функция возвращает false, иначе — true.

Функция findCommonIdenticalDigitNumbers находит пересечение двух множеств с помощью цикла. Для каждого числа из пересечения вызывается функция hasIdenticalDigits. Если число состоит из одинаковых цифр, оно добавляется в результирующее множество. Пересечение производится с использованием метода count, который проверяет наличие элемента во втором множестве.

Множество A: {111, 222, 333}
Множество B: {111, 444, 555}
Результат: Пересечение с одинаковыми цифрами: 111

Множество A: {1, 2, 3, 4, 5}
Множество B: {6, 7, 8, 9, 10}
Результат: Пересечение с одинаковыми цифрами: (пусто)

Множество A: {111, 123, 999}
Множество B: {111, 555, 999}
Результат: Пересечение с одинаковыми цифрами: 111 999