Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Южно-Уральский государственный университет (НИУ)»

Высшая школа электроники и компьютерных наук

Кафедра системного программирования

ОТЧЕТ

о выполнении практического задания №3

по дисциплине

«Структуры и алгоритмы обработки данных»

Вариант 5

Проверил:

ст. преподаватель кафедры СП

Петрова Л.Н.

Выполнил:

Студент группы КЭз-391

Галиулин Р.Р.

Челябинск
2025

Содержание

1. Описание задачи

Задание №1: Множество

Составить программу подсчёта общего количества цифр и знаков «+»,«-»,«*» в строке, введённой с клавиатуры.

Входные данные

- Строка, содержащая число-буквенные символы

Все данные вводятся с помощью стандартного потока вывода.

Выходные данные

- Количество символов «+»,«-»,«*» и цифр выведенных во входной строке. Целое положительное число или ноль.

Все данные вводятся с помощью стандартного потока вывода.

Задание №2: Очередь

Дана очередь вещественных чисел. Удалить из очереди числа из заданного пользователем диапазона.

Входные данные

- Очередь вещественных чисел
- Нижняя граница диапазона, вещественное число
- Верхняя граница диапазона, вещественное число

Все данные вводятся с помощью стандартного потока вывода.

Выходные данные

- Очередь вещественных чисел

Все данные вводятся с помощью стандартного потока вывода.

2. Листинги программ

Язык программирования: C++ 14. Среда разработки: Ubuntu 24.10, gcc 14.2.0, nvim

Листинг 1: Задание 1: Множество

```cpp
// Галиулин РР.. КЭз-391
// Структуры и алгоритмы обработки данных
// Практическое занятие №3

// Составить программу подсчета общего количества цифр и знаков «»«»«»+,-,*
// в строке, введенной с клавиатуры.

#include <iostream>
#include <string>
#include <set>

int main() {
    // Определяем множество символов, которые нужно учитывать
    std::set<char> symbols = {'0', '1', '2', '3', '4', '5',
                              '6', '7', '8', '9', '+', '-', '*'};

    // Ввод строки пользователем
    std::string input;
    std::cout << "Введите строку: ";
    std::getline(std::cin, input);

    // Счетчик для хранения общего количества найденных символов
    int count = 0;

    // Перебор каждого символа строки
    for (char c : input) {
        if (symbols.count(c) > 0) { // Проверяем, есть ли символ в множестве
            ++count;
        }
    }

    // Вывод результата
    std::cout << "Общее количество цифр и знаков '+','-','*': "
              << count << std::endl;

    return 0;
}
```

Листинг 2: Задание 2: Очередь

```cpp
// Галиулин РР.. КЭз-391
// Структуры и алгоритмы обработки данных
// Практическое занятие №3

// Дана очередь вещественных чисел. Удалить из очереди числа из заданного
// пользователем диапазона.

#include <iostream>
#include <queue>
#include <sstream>
#include <string>
#include <limits>

int main() {
    std::queue<double> numbers_queue;
    std::string input;
```

```
17      double number;
18
19      // Ввод данных единой строкой
20      std::cout << "Введите числа для очереди через пробел нажмите( Enter для"
21                   "завершения ввода): ";
22      std::getline(std::cin, input);
23      std::istringstream iss(input);
24
25      while (iss >> number) {
26          numbers_queue.push(number);
27      }
28
29      // Вывод очереди перед удалением
30      std::queue<double> copy_numbers_queue = numbers_queue;
31      std::cout << "Очередь перед удалением: ";
32      while (!copy_numbers_queue.empty()) {
33          std::cout << copy_numbers_queue.front() << " ";
34          copy_numbers_queue.pop();
35      }
36      std::cout << std::endl;
37
38      // Ввод диапазона
39      double lower_bound, upper_bound;
40      std::cout << "Введите нижнюю границу диапазона: ";
41      std::cin >> lower_bound;
42      std::cout << "Введите верхнюю границу диапазона: ";
43      std::cin >> upper_bound;
44
45      // Удаление элементов из очереди
46      std::queue<double> filtered_mumbers_queue;
47      while (!numbers_queue.empty()) {
48          double current = numbers_queue.front();
49          numbers_queue.pop();
50          if (current < lower_bound || current > upper_bound) {
51              filtered_mumbers_queue.push(current);
52          }
53      }
54
55      // Вывод результата
56      std::cout << "Очередь после удаления чисел из заданного диапазона: ";
57      while (!filtered_mumbers_queue.empty()) {
58          std::cout << filtered_mumbers_queue.front() << " ";
59          filtered_mumbers_queue.pop();
60      }
61
62      return 0;
63  }
```

3. Контрольные тесты

3.1 Задание №1: Множества

Исходные данные	**Результат**
A+4+5+6dfg56+-H41k12	14
0123456789g	10
1gfhgjhg5	2
fhghjgj2hfhgjhg	1

Таблица 1: Таблица с результатами контрольных тестов Задания №1

3.2 Задание №3: Очередь

Исходные данные	**Результат**
12 0 12 01 45 2 3 12 12	0 1 45 2 3
1 2 3 4 5 6 7 8 9 0 0 4	5 6 7 8 9
4 4 2 2 3 3 5 5 6 6 1 1 1 2	4 4 3 3 5 5 6 6

Таблица 2: Таблица с результатами контрольных тестов Задания №2

4. Контрольные вопросы

4.1 Дайте определение очереди

В отличии от стека очередь реализует принцип FIFO - первым пришёл, первым вышел, т.е. элементы добавляются в конец очереди, а извлекаются из начала.

4.2 Где применяется очередь?

Где логически применим принцип FIFO. Например при буферизации данных используется в качестве временного хранилища данных поступающих с разной скоростью. Управлении задачами, когда важно выполнять все по очереди, например очередь печати принтера, очереди сообщений в операционных системах, обработка запросов к серверу, моделирование процессов В ряде алгоритмов работы с более сложными структурами данных, например графами.

4.3 Перечислите основные операции применяемые в очереди.

Основными операциями являются:

- **Push** - добавление элемента в конец
- **Pop** - извлечение элемента с начала

Также существуют, но не во всех реализациях, дополнительные операции:

- **Top** (или Peek) - просмотр начального элемента без извлечения
- Проверка на пустоту, наличия хотя-бы одного элемента.
- **Size** - размер стека

4.4 Что такое множество?

В качестве структуры данных, являются отражением математического понятия, как неупорядоченную коллекцию различных элементов. Ключевым является что элементы - различны, т.е. множество не содержит в себе дубликаты.

4.5 Дайте определение мощности множества.

Кол-во **различных** элементов множества. Если множество реализуется с соблюдением принципа различности элементов, то просто кол-во элементов множества.

4.6 Какие операции можно применять к множествам?

- Добавление элемента в множество

- Удаление элемента из множества
- Проверка на принадлежность некого значения множеству
- Объединение (аналог $Q \cup W$) - создание нового множества состоящего из элементов множеств подлежащих объединению (в std::set set_union)
- Пересечение (аналог $Q \cap W$) - создание нового множества состоящих из элементов принадлежащих (состоящих) в множествах подлежащих пересечению (в std::set set_intersection)
- Разность множеств (аналог Q-W) - создание нового множества элементы которого принадлежат Q но не принадлежат W (в std::set set_difference)
- Проверка на подмножество (аналог $Q \subset W$) - логическая операция возвращающая истину если все элементы множества Q также содержаться в множестве W