МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ФАКУЛЬТЕТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК ИМЕНИ
ПРОФЕССОРА Н.И.ЧЕРВЯКОВА

ЛАБОРАТОРНАЯ РАБОТА №19

Алгоритмизация и программирование

Множества

Выполнил студент:
Сивко Иван Андреевич
студент 2 курса
группа ПМИ-б-о-23-2,
направление подготовки 01.03.02

Проверил:
Ассистент кафедры вычислительной
математики и кибернетики, к.ф.-м.н.,
Черкашина Анастасия Андреевна

2024 г.

Вариант 9

Цель:

- Совершенствование навыков разработки программ в среде программирования MS Visual Studio
- Совершенствование навыков в программировании с использованием множеств
- Исследование процесса формирования множеств
- Исследование операций с элементами множеств

Задание 1

1 Условие:

Дан текст из строчных латинских букв, за которым следует точка. Вывести на экран все буквы входящие в текст по одному разу.

2 Алгоритм / Мат. модель

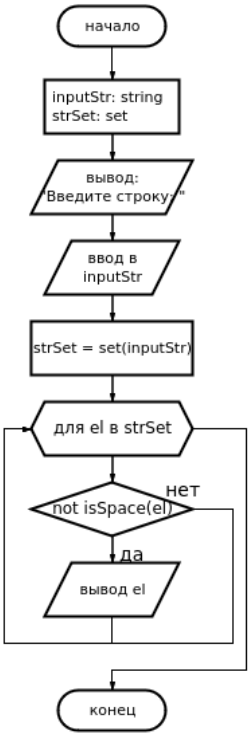
Программа анализирует строку, состоящую из строчных латинских букв, завершающуюся точкой, и выводит все уникальные символы этой строки, исключая пробелы. Для этого используется структура данных, обеспечивающая хранение уникальных элементов (в данном случае — множества). Алгоритм работает следующим образом:

1. **Чтение входных данных:**
   Программа сначала проверяет наличие аргументов командной строки. Если они есть, строка собирается из всех переданных аргументов. В противном случае программа запрашивает строку у пользователя через стандартный ввод, ограничивая ввод точкой, чтобы исключить символы после неё.
2. **Использование множества для хранения уникальных символов:**
   Весь текст из строки помещается в стандартное множество `std::set`, которое автоматически удаляет все дубликаты, оставляя только уникальные символы.
3. **Фильтрация пробелов:**
   При выводе уникальных символов из множества проверяется, что символ не является пробелом. Это делается с помощью стандартной функции `isspace`.
4. **Вывод результата:**
   Уникальные символы выводятся на экран через пробел.
5. **Завершение работы программы:**
   После выполнения всех операций программа корректно завершает выполнение, возвращая код `EXIT_SUCCESS`.

Название	**Тип**	**Описание**
Переменные main		
inputStr	std::string	Строка, в которой будет храниться весь ввод пользователя или аргументы командной строки.
chr	char	Переменная, которая используется для итерации по символам строки в процессе вывода уникальных символов.

Таблица 1: Переменные, функции и структуры, используемые в программе

3 Диаграмма:



4 Код:

```
#include <cstdlib>
#include <cstdint>
#include <cctype>
#include <set>
#include <limits>
#include <iostream>

int main(int32_t argc, const char** argv) {
    std::string inputStr{};
    if (argc > 1)
        for (size_t i{1}; i<argc; ++i)
            inputStr += argv[i];
    else {
        std::cout << "Введите строку: ";
        std::getline(std::cin >> std::ws, inputStr, '.');
        std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
    }
    std::cout << "Уникальные символы в строке: ";
    for (char chr : std::set<char>(inputStr.begin(), inputStr.end()))
        if (!isspace(chr))
            std::cout << chr << ' ';
    std::cout << '\n';
    return EXIT_SUCCESS;
}
```

source code

5 Результат работы программы:

```
$ ./a.out
ВВедите строку: Hello world
hello xd xd xd
Уникальные символы в строке: H d e h l o r w x
$ ./a.out "Hello world"
Уникальные символы в строке: H d e l o r w
```

Задание 2

Выбрав произвольные множества A, B и C с ограничениями C++, проиллюстрировать истинность следующих свойств операций над множествами и отношений между ними (с нижним индексом в формулах используется дополнение множества до соответствующего базового типа T, т.е. Bt = T – B, где B : set of T).

1 Условие:

Выбрав произвольные множества A, B и C с ограничениями C++, проиллюстрировать истинность следующих свойств операций над множествами и отношений между ними (с нижним индексом в формулах используется дополнение множества до соответствующего базового типа T, т.е. Bt = T – B, где B : set of T).

1. $A \setminus B = A \cap B_T$;
2. $A \setminus (A \setminus B) = A \cap B$;
3. $B \cup (A \setminus B) = A \cup B$;
4. $B \cap (A \setminus B) = \varnothing$;
5. $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$;
6. $A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$;
7. $A \cap B \subset A \cup B$;
8. $A = (A \setminus B) \cup (A \cap B)$;
9. $(A \setminus B) \cap (A \cap B) = \varnothing$;
10. $(A \setminus B) \cap B = \varnothing$;
11. $A \setminus B = A \setminus (A \cap B)$;
12. $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$;
13. $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$;
14. $(A \setminus B) \setminus C = (A \setminus C) \setminus B$;
15. $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$;
16. $A \cup (B \setminus C) \supset (A \cup B) \setminus C$;
17. $(A \setminus B) \cup C \supset (A \cup C) \setminus B$;
18. $(A \setminus B)_T = A_T \cup (A \cap C)$;
19. $A \cup (B \cup C) = (A \cup B) \cup C$;
20. $A \cap (B \cap C) = (A \cap B) \cap C$;
21. $(A \cap B_T)_T = A_T \cup B$;
22. $(A \cup B)_T = A_T \cap B_T$;
23. $(A \cap B)_T = A_T \cup B_T$;
24. $A \cap (B \setminus C) = (A \cap B) \setminus (A \cap C)$;
25. $A \setminus (B \cup C) = (A \setminus B) \setminus C$.

2 Алгоритм / Мат. модель

Программа выполняет операции над двумя множествами целых чисел, введёнными пользователем. После ввода данных выполняются следующие шаги:

1. **Инициализация переменных:**
   Программа инициализирует несколько переменных:
   - `setA` и `setB` — множества типа `uint32_t`, которые будут хранить элементы, введённые пользователем.
2. **Ввод данных:**
   - Пользователь вводит элементы для множества `setA`. Ввод продолжается до тех пор, пока не будет введена пустая строка или некорректное значение.

- Ввод элементов множества `setB` аналогичен.
- Для каждого введённого числа функция `operator»` пытается преобразовать строку в число и добавить его в соответствующее множество.
- Если ввод некорректен (например, введено нечисловое значение), программа выведет сообщение об ошибке и запросит ввод снова.

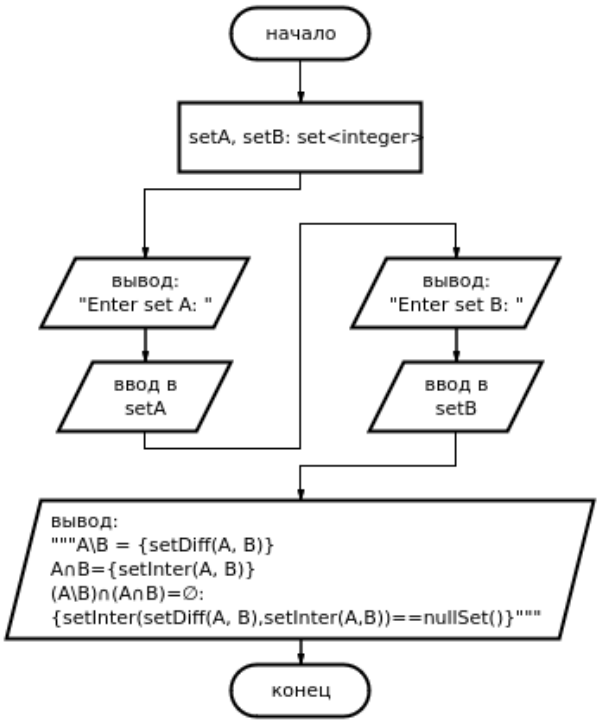
3. **Операции над множествами:**
   - Программа выполняет операцию вычитания множества $A \setminus B$, используя перегруженный `operator-`. Результат выводится на экран.
   - Затем выполняется операция пересечения множеств $A \cap B$ с использованием перегруженного `operator&`. Результат также выводится на экран.
   - Далее проверяется, является ли пересечение множества $A \setminus B$ и множества $A \cap B$ пустым с использованием операции `operator&`. Результат выводится в виде логического значения `true` или `false`.

4. **Вывод результатов:**
   - Программа выводит результат операции $A \setminus B$, затем результат операции $A \cap B$.
   - В конце выводится результат проверки условия $(A \setminus B) \cap (A \cap B) = \varnothing$, которое возвращает логическое значение, подтверждающее или опровергающее пустоту пересечения.

Название	**Тип**	**Описание**
Функции		
operator»()	std::istream&	Перегрузка оператора ввода для ввода множества. Читает числа из потока и добавляет их в множество.
operator«()	std::ostream&	Перегрузка оператора вывода для вывода множества в стандартный поток вывода.
operator-()	std::set<T>	Перегрузка оператора вычитания для множеств, возвращающая элементы первого множества, которые не принадлежат второму.
operator&()	std::set<T>	Перегрузка оператора пересечения для множеств, возвращающая элементы, общие для двух множеств.
Переменные main		
setA	std::set<uint32_t>	Множество для хранения элементов первого множества, введенного пользователем.
setB	std::set<uint32_t>	Множество для хранения элементов второго множества, введенного пользователем.

Таблица 2: Переменные, функции и типы данных, используемые в программе

3 Диаграмма:



4 Код:

```
#include <cstdlib>
#include <cstdint>
#include <set>
#include <sstream>
#include <algorithm>
#include <iostream>

std::istream& operator>>(std::istream& in, std::set<uint32_t>& set) {
    std::string inputStr;
    while (true) {
        std::getline(in, inputStr);
        std::stringstream sstream(inputStr);
        if (uint32_t num; sstream >> num)
            set.insert(num);
        else if (!inputStr.empty())
            std::cout << "Wrong input, try again\n";
        else
            break;
    }
    return in;
}

template <typename T>
std::ostream& operator<<(std::ostream& out, const std::set<T>& set) {
    for (const T& el : set)
        out << el << ' ';
```

```
    return out;
}

template <typename T>
std::set<T> operator-(const std::set<T>& a, const std::set<T>& b) {
    std::set<T> res;
    std::set_difference(a.begin(), a.end(), b.begin(), b.end(), std::inserter(res, res.end
    return res;
}

template <typename T>
std::set<T> operator&(const std::set<T>& a, const std::set<T>& b) {
    std::set<T> res;
    std::set_intersection(a.begin(), a.end(), b.begin(), b.end(), std::inserter(res, res.e
    return res;
}

int main() {
    std::set<uint32_t> setA, setB;
    std::cout << "Enter set A: ";
    std::cin >> setA;
    std::cout << "Enter set B: ";
    std::cin >> setB;

    std::cout << "A - B: " << (setA - setB) << '\n'
        << "A & B: " << (setA & setB) << '\n'
        << "(A \\ B) & (A & B)=={}: "
        << std::boolalpha << (((setA - setB) & (setA & setB)) == std::set<uint32_t>{}) <<
    return EXIT_SUCCESS;
}
```

source code

5 Результат работы программы:

```
$ ./a.out
Enter set A: 1
2

Enter set B: 2
3

A - B: 1
A & B: 2
(A \ B) & (A & B)=={}: true
```