Лабораторная работа 2: последовательные контейнеры и алгоритмы

Структура проекта:

task02-sequence-containers

|-- .clang-format [опционально]

|-- .gitignore

|-- .gitlab-ci.yml

|-- Makefile

|-- soundex.cpp

`-- yahtzee.cpp

Цель работы: После выполнения команды make должны создаваться два исполняемых файла: soundex и yahtzee.

**Задание 1: yahtzee**

Яцзы — это покер на костях, в игре используется пять кубиков. Ваша задача — реализовать программу, которая по заданной комбинации вычисляет количество очков в каждой категории.

| **Категория** | **Описание** | **Количество очков** | **Пример** |
| --- | --- | --- | --- |
| Ones |  | Сумма всех единиц | 1 1 1 3 4 => 3 |
| Twos |  | Сумма всех двоек | 2 2 2 5 6 => 6 |
| Threes |  | Сумма всех троек | 3 3 3 3 4 => 12 |
| Fours |  | Сумма всех четверок | 4 4 5 5 5 => 8 |
| Fives |  | Сумма всех пятерок | 1 1 2 2 5 => 5 |
| Sixes |  | Сумма всех шестерок | 3 3 6 6 6 => 18 |
| Three Of A Kind | Минимум три одинаковых значения | Сумма всех значений | 2 3 4 4 4 => 17 |
| Four Of A Kind | Минимум четыре одинаковых значения | Сумма всех значений | 4 5 5 5 5 => 24 |
| Full House | Пара + тройка | 25 | 2 2 5 5 5 => 25 |
| Small Straight | Четыре последовательных значения | 30 | 1 3 4 5 6 => 30 |
| Large Straight | Пять последовательных значений | 40 | 1 2 3 4 5 => 40 |
| Yahtzee | Все значения равные | 50 | 3 3 3 3 3 => 50 |
| Chance | Любая комбинация | Сумма всех значений | 1 1 3 4 5 => 14 |

Приложение принимает на вход пять значений от 1 до 6 в произвольном порядке. Если хотя бы одно из значений не входит в заданный диапазон, приложение завершается с ненулевым кодом.

При реализации используйте последовательные контейнеры и алгоритмы стандартной библиотеки C++ на свой выбор:

* std::array
* std::vector
* <algorithm>

Пример запуска приложения:

$ ./yahtzee

1 1 1 1 1

Ones: 5

Twos: 0

Threes: 0

Fours: 0

Fives: 0

Sixes: 0

Three Of A Kind: 5

Four Of A Kind: 5

Full House: 0

Small Straight: 0

Large Straight: 0

Yahtzee: 50

Chance: 5

$ ./yahtzee

7 1 1 1 1

Error: invalid dice value

**Задание 2: soundex**

Soundex — фонетический алгоритм, возвращающий одинаковые значения ключа для схожих по звучанию имен в английском языке.

Напишите утилиту soundex, реализующую следующий алгоритм:

1. Запоминается первая буква слова.
2. Удаляются все вхождения h и w (за исключением первой буквы слова).
3. Согласные заменяются на цифры от 1 до 6, причём похожим по звучанию буквам соответствуют одинаковые цифры:
   * b, f, p, v → 1
   * c, g, j, k, q, s, x, z → 2
   * d, t → 3
   * l → 4
   * m, n → 5
   * r → 6
4. Любая последовательность одинаковых цифр сокращается до одной такой цифры.
5. Удаляются все a, e, i, o, u, y (за исключением первой буквы слова).
6. Заменяется первый символ буквой, запомненной на шаге 1, делая её заглавной.
7. Итоговая строка обрезается до первых четырёх символов. Если длина строки меньше требуемой, недостающие символы заменяются знаком 0.

Если слово содержит символы кроме [a-zA-Z], то вернуть 0000.

При реализации алгоритма не пишите циклы, по максимуму пользуйтесь стандартными алгоритмами: Algorithms library.

Допускается использование как unconstrained алгоритмы, так и алгоритмы из пространства имен std::ranges.

Синтаксис запуска:

soundex <name>

Пример запуска приложения:

$ ./soundex Aya

A000

$ ./soundex 42

0000