Изображение выглядит как текст, небо, внешний

Автоматически созданное описание

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**   
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**   
**імені ІГОРЯ СІКОРСЬКОГО»**

**Інститут прикладного системного аналізу**   
**Кафедра системного проектування**  
   
   
 

**Звіт**

**про виконання лабораторної роботи №12**   
**«****Робота з рядками в С++»**

Виконав:   
студент I курсу, групи ДА-12   
Панасенко Олександр Андрійович

Прийняв:

к.т.н., доцент Безносик О. Ю.

**Мета роботи**

вивчити синтаксис і семантику стандартних методів по роботі з рядками, навчитися використовувати в програмах методи для роботи з рядками на мові C++.

**Завдання**

1. Зашифруйте речення, що вводиться з клавіатури, у такий спосіб: спочатку вибираються два довільних слова з бази, що знаходиться в тексті програми або вводиться з клавіатури, потім слово із речення, що шифрується, потім знову два слова з бази, після чого - знову слово з речення і т.д. "База" - набір слів, припустимих при виконанні програми, або набір пар слів, як у приведеному нижче прикладі.  
2. Напишіть дешифратор. Можливі символи - прописні українські букви; символ-роздільник '\_'.

**Приклад:**  
Речення, що шифрується:  
*СПРАВА\_ЗАКІНЧЕНА\_ХАДСОН\_РОЗПОВІВ\_ВСЕ\_СТЕРЕЖІТЬСЯ*  
База шифру:  
*З\_ДИЧИНОЮ\_Я\_ВВАЖАЮ\_ГОЛОВА\_ПІДПРИЄМСТВА\_ЗА\_ВІДОМОСТЯМИ\_ПРО\_МУХОБІЙКІ\_*  
ФАЗАНЬЇХ\_КУРОЧОК  
Результат шифровки:  
*З\_ДИЧИНОЮ\_СПРАВА\_Я\_ВВАЖАЮ\_ЗАКІНЧЕНА\_ГОЛОВА\_ПІДПРИЄМСТВА\_ХАДСОН\_ЗА\_ВІДОМОСТЯМИ\_РОЗПОВІВ\_ПРО\_МУХОБІЙКІ\_ВСЕ\_ФАЗАНЬЇХ\_КУРОЧОК\_СТЕРЕЖІТЬСЯ*

**Аналіз умови задачі.**

Виходячи з умови задачі, я створив 3 функції, для шифровки, дешифровки та ділення речення на слова. Ділення речення на слова я реалізував за допомогою класа std::vector та функції strtok(), а результат шифровки та дешифровки я отримував з використанням методів класу std::string, такі як append() та c\_str().

Ввод користувача я опрацьовував потоком std::cin, а не fgets(), адже у вводі не використовуються пробіли.

**Код програми**

#include <iostream>

#include <vector>

#include <string.h>

// will split a string by delimeter and create a vector

std::vector<std::string> split\_delim(std::string str, char delim)

{

std::vector<std::string> m\_words;

const char\* cstr = str.c\_str();

char\* word = strtok((char\*) cstr, &delim); // <- (char\*) cast is very unlikely as it may be UB, but not for strtok (no modification of string)

while (word != NULL)

{

m\_words.push\_back(word);

word = strtok(NULL, &delim); // <- will point at next char after delim

}

return m\_words;

}

std::string encode(const char\* base, const char\* input, char delim)

{

std::vector<std::string> base\_words = split\_delim(base, delim);

std::vector<std::string> input\_words = split\_delim(input, delim);

// Base: HELLO\_WORLD\_GOODBYE\_NIGHT

// Input: HI\_GUYS

// BASE\_BASE\_INPUT\_BASE\_BASE\_INPUT

// Output: HELLO\_WORLD\_HI\_GOODBYE\_NIGHT\_GUYS

size\_t word\_count = base\_words.size() + input\_words.size();

// int act\_size = input\_size + base\_size + 1; // 1 stands for extra delimeter after concatenating both strings

std::string result\_string;

int input\_cursor = 0;

int base\_cursor = 0;

if (base\_words.size() / 2 != input\_words.size()) // check if there are enough words in BASE, so there are no std::out\_of\_range error

throw std::range\_error("Not enough words in base or input!");

for (size\_t i = 0; i < word\_count; i++)

{

std::string target = i % 3 == 2 ? input\_words.at(input\_cursor++) : base\_words.at(base\_cursor++);

result\_string.append(target);

if (i != word\_count - 1)

result\_string.append(&delim);

}

// result\_string[act\_size] = '\0'; // <- null-char automatically appended by std::string

return result\_string;

}

std::string decode(std::string buffer, char delim)

{

std::vector<std::string> buffvec = split\_delim(buffer.c\_str(), delim);

std::string decoded;

size\_t word\_count = buffvec.size();

for (int i = 0; i < word\_count; i++)

{

if (i % 3 == 2)

{

std::string curr\_word = buffvec.at(i);

decoded.append(curr\_word);

if (i != word\_count - 1)

decoded.append(&delim);

}

}

// newbuff[buff\_size] = '\0'; // <- null-term char automatically appended by std::string

return decoded;

}

int main(void)

{

char delim = '\_';

std::string input;

std::string base;

std::cout << "Enter base**\n**";

std::cin >> base;

std::cout << "[LOG] Saved Base: " << base << std::endl;

std::cout << "Enter input**\n**";

std::cin >> input;

std::cout << "[LOG] Saved Input: " << input << std::endl;

std::string encoded = encode(base.c\_str(), input.c\_str(), delim);

std::cout << "[LOG] Encoded result: " << encoded << std::endl;

std::cout << "[LOG] Decoded result: " << decode(encoded, delim) << std::endl;

}

Результати виконання програми зображено на рисунку 1.

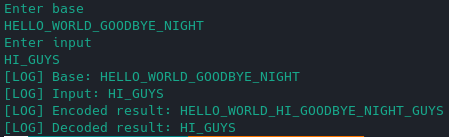


Рисунок 1

**Висновки**

У ході виконання лабораторної роботи №12 я використовував об’єкти класу std::string, та його методи для опрацювання вводу користувача, який я зчитував потоком std::cin. Для ділення речення на слова я використав функцію мови C — strtok(). Отримані слова я зберігав у структуру даних std::vector, що є динамічним масивом.