Отчет по лабораторной работе № 13 по курсу “Фундаментальная информатика”

Студент группы М80-109Б-22 Федоров Алексей Алексеевич, № 20
Контакты:
Email: hedgefog@yandex.ru, Telegram: @hedgefo9

Работа выполнена: «18» декабря 2022г.

Преподаватель: каф. 806 Сысоев Максим Алексеевич

Отчет сдан « » _____20__ г., итоговая оценка _____

Подпись преподавателя

_____

1. **Тема:** Множества
2. **Цель работы:** Составить программу на языке Си, выполняющую работу со множествами
3. **Задание (Вариант №18)**
   Проверить, есть ли два соседних слова с непересекающимся набором согласных во входных данных?
4. **Оборудование (студента):**
   Процессор *Ryzen 5 3500U @ 8x 2.1GH* с ОП *14900* Мб, НМД *677* Гб. Монитор *2160x1440*
5. **Программное обеспечение (студента):**
   Операционная система семейства: *linux*, наименование: *ubuntu*, версия *18.10 cinnamon* интерпретатор команд: *bash* версия *4.4.19*.
   Система программирования -- версия --, редактор текстов *emacs* версия *25.2.2*
6. **Идея, метод, алгоритм**
   Идея заключается в том, чтобы посимвольно добавлять буквы 2 слов в 2 множества. Флаг изначально равен 0. Затем нужно искать пересечение трёх множеств: множества букв 1 слова, множества букв 2 слова и множества согласных. Если пересечение равно пустому множеству (т. е. 0), то флаг сделать равным 1. Затем нужно в первое множество записать значение из второго множества, а во второе множество записать 0 и посимвольно добавить в него следующее слово. Если после прохода всех входных данных флаг равен 1, то нужно вывести фразу о том, что во входных данных есть два соседних слова с непересекающимся набором согласных, иначе – что нет.
7. **Сценарий выполнения работы**

Тесты

Входные данные	Выходные данные	Результат работы программы	Вердикт
aaa bbbbb acacacb	Input contains at least two adjacent words with disjoint set of consonants	Input contains at least two adjacent words with disjoint set of consonants	OK
cake panic clown circus	Input does not contain at least two adjacent words with disjoint set of consonants	Input does not contain at least two adjacent words with disjoint set of consonants	OK

Также проведены Unit-тесты внутри программы

8. **Распечатка протокола:**

Код на языке C

```c
#include <stdio.h>
#include <ctype.h>
#include <assert.h>

#define ALL_LETTERS 67108863u
#define VOWELS (1u << ('a' - 'a') | 1u << ('e' - 'a') | 1u << ('i' - 'a') | 1u << ('o' - 'a') | 1u << ('u' - 'a'))
#define CONSONANTS ((~VOWELS) & ALL_LETTERS)

typedef enum {
```

```
  SEPARATOR,
  WORD1,
  WORD2
} state;

unsigned int char_to_set(char c) {
    c = tolower(c);
    if (c < 'a' || c > 'z') {
        return 0u;
    } else {
        return 1u << (c - 'a');
    }
}
int is_separator(char c) {
    return c == ' ' || c == '\n' || c == '\t' || c == ',' || c == EOF;
}

int is_letter(char c) {
    return (char_to_set(c) & ALL_LETTERS) > 0;
}

void test_char_to_set() {
    assert(char_to_set('a') == 1u);
    assert(char_to_set('y') == 16777216u);
    assert(char_to_set('M') == 4096u);
    assert(char_to_set('+') == 0u);
}

void test_is_separator() {
    assert(is_separator(' ') == 1);
    assert(is_separator('\t') == 1);
    assert(is_separator(EOF) == 1);
    assert(is_separator('f') == 0);
}

void test_is_letter() {
    assert(is_letter('a') == 1);
    assert(is_letter('y') == 1);
    assert(is_letter('M') == 1);
    assert(is_letter('+') == 0);
}

void test_all() {
    test_char_to_set();
    test_is_separator();
    test_is_letter();
}

int main() {
    test_all();
    int result = 0;
    char c = ' ';
    state current_state = SEPARATOR;
    unsigned int word1_set = 0u, word2_set = 0u;

    while (1) {
        c = getchar();
        switch (current_state) {
            case SEPARATOR:
                if (is_letter(c)) {
                    word1_set = char_to_set(c);
                    current_state = WORD1;
                }
                break;
            case WORD1:
                if (is_letter(c)) {
                    word1_set = word1_set | char_to_set(c);
                } else if (is_separator(c)) {
                    current_state = WORD2;
```

```
                }
                break;
            case WORD2:
                if (is_letter(c)) {
                    word2_set = word2_set | char_to_set(c);
                } else if (is_separator(c)) {
                    if ((word1_set > 0) && (word2_set > 0)) {
                        if (((word1_set & word2_set) & CONSONANTS) == 0u) {
                            result = 1;
                        }
                        word1_set = word2_set;
                        word2_set = 0;
                    }
                }
                break;
        }
        if (c == EOF) break;
    }

    if (result == 1) {
        printf("Input contains at least two adjacent words with disjoint set of
consonants\n");
    } else {
        printf("Input does not contain at least two adjacent words with disjoint set of
consonants\n");
    }

    getchar();
    return 0;
}
```

9. **Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. **Замечания автора**

Нет замечаний

11. **Выводы**

Благодаря этой лабораторной работе я потренировался работать с множествами. Логика проста и понятна. Мне понравилась данная работа, т. к. я познакомился с одной из абстракций для представления множества в Си. Конечно, реальные множества (как структура данных, например в C++) устроены сложнее и в их основе лежат более тяжёлые алгоритмы (например, с помощью «красно-чёрных деревьев»), но к их пониманию нужно приходить пошагово, так что это хорошая первая ступень.

Подпись студента _____