|  |  |
| --- | --- |
|  | **Отчёт по лабораторной работе** № 13  по курсу: Фундаментальная информатика  студента группы : M8O-105Б-21 Козлова Егора Сергеевича , № по списку: 10  Контакты: e-mail iamaghoulzxc@gmail.com  Работа выполнена: “3“ декабря 2021 г.  Преподаватель: каф. 806 В. К. Титов  Входной контроль знаний с оценкой  Отчёт сдан: “4“ декабря 2021 г., итоговая оценка  Подпись преподавателя |

1. **Тема**: Множества. Массивы. Хедер файлы.
2. **Цель работы**: написать на языке Си программу проверки характеристик введенных последовательностей слов.
3. **Задание** (*вариант №* \_ ):

На ввод поступают слово-образец и строка. Необходимо вывести наиболее длинное слово из вводимой строки, множество букв которого являются подмножеством множества букв вводимого слова-образца. Если таких слов несколько, то вывести любое из них.

\*гарантируется, что максимальная длина слова не превышает 63 букв

1. **Оборудование** (*лабораторное*):

*Не использовалось*

*Оборудование ПЭВМ студента, если использовалось:*

Процессор AMD Ryzen 7 5800H @ 8x 3.2GHz , ОП 15429 МБ, НМД 1024 ГБ. Монитор: встроенный (1920x1080)

1. *Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства Linux, наименование: Ubuntu версия 20.04.3 LTS x86\_64 \_\_

Интерпретатор команд: bash версия 5.0.17

Редактор текстов: Emacs версия 26.3

Утилиты операционной системы:

Прикладные системы и программы: gcc

Местонахождения и имена файлов программ и данных: /bin

1. **Идея, метод, алгоритм** решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальное описание с пред- и постусловиями)

//word.h

Для выполнения данной задачи напишем несколько функций, а именно: функцию очистки символьного массива, функцию, которая считает длину слова, функцию, которая копирует символьный массив.

Функция очистки символьного массива (Clear): на вход подается символьный массив. Проходимся циклом по этому массиву, заменяя каждый элемент на \0.

Функция, вычисляющая длину слова (Len): на вход подается символьный массив. Проходимся циклом по этому массиву. Если i-тый элемент является буквой, то увеличить счетчик на 1.

Функция, копирующая символьный массив (Copy): На вход подаются два массива. Очищаем второй массив с помощью функции Clear. В цикле присваиваем i-тому элементу второго массива i-тый элемент первого.

//13.c

Пока считываемый символ не является концом файла выполняем:

Обнуляем итератор. Очищаем массив input. Запускаем цикл, в котором считываем символ и если этот символ является буквой (Letter), то присваиваем его значение i-тому элементу массива input. Увеличиваем итератор на единицу.

Если это был первый ввод (флаг поднят, first = 1), значит ввели слово-образец, следовательно создаем множество букв введенного слова (CreateSet) и опускаем флаг (first = 0) .

Вычисляем длину введенного слова (Len) и создаем множество его букв (CreateSet). Если длина этого слова больше максимальной длины, найденной на данный момент, и множество его букв является подмножеством множества букв слова-образца, то запоминаем это слово в массив ans путем копирования (Copy).

По завершении цикла выводим массив ans, который и будет являться ответом.

1. **Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты, либо соображения по тестированию].

//13.c  
  
#include <stdio.h>  
#include "headers/set.h"  
#include "headers/word.h"  
  
int main() {  
 char c;  
 char input[**LEN**] = {'\0'}, example\_set[**N** \* 2], set[**N** \* 2], ans[**LEN**] = {'\0'};  
 int len, maxlen = 0, first = 1, i;  
  
 while (c != **EOF**) {  
 i = 0;  
 Clear(input);  
 while (((c = getchar()) != ' ') && (c != '\0') && (c != '\n')) {  
 if (Letter(c)) input[i] = c;  
 else break;  
 ++i;  
 }  
 if (first) {  
 first = 0;  
 CreateSet(input, example\_set);  
 continue;  
 }  
 len = Len(input);  
 CreateSet(input, set);  
 if ((len > maxlen) && (IncludeSet(set, example\_set))) {  
 Copy(input, ans);  
 maxlen = len;  
 }  
 }  
 printf("%s\n", ans);  
 return 0;  
}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

// word.h  
  
#define **LEN** 64  
  
void Clear(char W[]) {  
 for (int i = 0; i < **LEN**; ++i) W[i] = '\0';  
}  
  
int Len(char W[]) {  
 int i;  
 for (i = 0; W[i] != '\0'; ++i) {}  
 return i;  
}  
  
void Copy(char W1[], char W2[]) {  
 Clear(W2);  
 for (int i = 0; i < Len(W1); ++i) W2[i] = W1[i];  
}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

// set.h  
  
#define **N** 26  
  
char A = 'A';  
char Z = 'Z';  
char a = 'a';  
char z = 'z';  
  
char Alph[2 \* **N**]; // alphabet: small and capital letters  
  
  
int Letter(char c) { return c >= a && c <= z || c >= A && c <= Z; }  
  
  
void CreateSet(char W[], char S[]) {  
 char c;  
 int i;  
 for (i = 0; i < **N**; i++)S[i] = S[**N** + i] = 0;  
 for (i = 0; c = W[i]; i++)  
 if (c >= a && c <= z) S[c - a] = 1;  
 else if (c >= A && c <= Z) S[**N** + c - A] = 1;  
}  
  
void DisplaySet(char S[]) {  
 int i;  
 Alph[0] = a;  
 Alph[**N**] = A;  
 Alph[2 \* **N**] = '\0'; // alphabet fills up  
 for (i = 1; i < **N**; i++) {  
 Alph[i] = Alph[i - 1] + 1;  
 Alph[i + **N**] = Alph[i + **N** - 1] + 1;  
 } // --"--  
 printf("\n%s\n", Alph); // and prints  
 for (i = 0; i < 2 \* **N**; i++)  
 printf("%1d", S[i]);  
 printf("\n"); // set prints  
}  
  
  
void UnionSet(char S1[], char S2[], char S3[]) { for (int i = 0; i < 2 \* **N**; i++) S3[i] = S1[i] || S2[i]; }  
  
void IntersSet(char S1[], char S2[], char S3[]) { for (int i = 0; i < 2 \* **N**; i++) S3[i] = S1[i] && S2[i]; }  
  
int IncludeSet(char S1[], char S2[]) {  
 for (int i = 0; i < 2 \* **N**; i++)  
 if (S1[i] == 1 && S2[i] == 0) return 0;  
 return 1;  
}  
  
int EmptySet(char S[]) {  
 for (int i = 0; i < 2 \* **N**; i++)  
 if (S[i] == 1) return 0;  
 return 1;  
}  
  
void InSet(char c, char S[]) // add to set  
{  
 if (Letter(c))  
 if (c <= z) S[c - a] = 1; else S[c + **N** - A] = 1;  
}

Тесты:

test1.txt

Входные данные:

example

exam is hard, but u can pass it. exxxxxaaaaammmmm, yeeeeeeaaaaaaaahhhhhhh

Выходные данные:

exxxxxaaaaammmmm

test2.txt

Входные данные:

Moscow

Msc moscow woooow how? 32425.

cow mosmosmosmos hey MosMosMosMos yo

Выходные данные:

MosMosMosMos

Допущен к выполнению работы. Подпись преподавателя

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем)

**isitmuse@isitmuse**:**~/lab/13**$ cat head13.txt

\_.\_ \_,-'""`-.\_

(,-.`.\_,'( |\`-/|

`-.-' \ )-`( , o o)

`- \`\_`"'-

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|||ЛАБОРАТОРНАЯ РАБОТА №8|||

|МНОЖЕСТВА, МАССИВЫ И ХЕДЕР||

||||||||||||ФАЙЛЫ||||||||||||

|||ВЫПОЛНИЛ СТУДЕНТ ГРУППЫ|||

|||М8О-105Б-21 КОЗЛОВ ЕГОР|||

\\\\\\\\\\\\\\|//////////////

**isitmuse@isitmuse**:**~/lab/13**$ cat 13.c

//13.c

#include <stdio.h>

#include "headers/set.h"

#include "headers/word.h"

int main() {

char c;

char input[LEN] = {'\0'}, example\_set[N \* 2], set[N \* 2], ans[LEN] = {'\0'};

int len, maxlen = 0, first = 1, i;

while (c != EOF) {

i = 0;

Clear(input);

while (((c = getchar()) != ' ') && (c != '\0') && (c != '\n')) {

if (Letter(c)) input[i] = c;

else break;

++i;

}

if (first) {

first = 0;

CreateSet(input, example\_set);

continue;

}

len = Len(input);

CreateSet(input, set);

if ((len > maxlen) && (IncludeSet(set, example\_set))) {

Copy(input, ans);

maxlen = len;

}

}

printf("%s\n", ans);

return 0;

}

**isitmuse@isitmuse**:**~/lab/13**$ cd headers/

**isitmuse@isitmuse**:**~/lab/13/headers**$ cat word.h

// word.h

#define LEN 64

void Clear(char W[]) {

for (int i = 0; i < LEN; ++i) W[i] = '\0';

}

int Len(char W[]) {

int i;

for (i = 0; W[i] != '\0'; ++i) {}

return i;

}

void Copy(char W1[], char W2[]) {

Clear(W2);

for (int i = 0; i < Len(W1); ++i) W2[i] = W1[i];

}

**isitmuse@isitmuse**:**~/lab/13/headers**$ cat set.h

// set.h

#define N 26

char A = 'A';

char Z = 'Z';

char a = 'a';

char z = 'z';

char Alph[2 \* N]; // alphabet: small and capital letters

int Letter(char c) { return c >= a && c <= z || c >= A && c <= Z; }

void CreateSet(char W[], char S[]) {

char c;

int i;

for (i = 0; i < N; i++)S[i] = S[N + i] = 0;

for (i = 0; c = W[i]; i++)

if (c >= a && c <= z) S[c - a] = 1;

else if (c >= A && c <= Z) S[N + c - A] = 1;

}

void DisplaySet(char S[]) {

int i;

Alph[0] = a;

Alph[N] = A;

Alph[2 \* N] = '\0'; // alphabet fills up

for (i = 1; i < N; i++) {

Alph[i] = Alph[i - 1] + 1;

Alph[i + N] = Alph[i + N - 1] + 1;

} // --"--

printf("\n%s\n", Alph); // and prints

for (i = 0; i < 2 \* N; i++)

printf("%1d", S[i]);

printf("\n"); // set prints

}

void UnionSet(char S1[], char S2[], char S3[]) { for (int i = 0; i < 2 \* N; i++) S3[i] = S1[i] || S2[i]; }

void IntersSet(char S1[], char S2[], char S3[]) { for (int i = 0; i < 2 \* N; i++) S3[i] = S1[i] && S2[i]; }

int IncludeSet(char S1[], char S2[]) {

for (int i = 0; i < 2 \* N; i++)

if (S1[i] == 1 && S2[i] == 0) return 0;

return 1;

}

int EmptySet(char S[]) {

for (int i = 0; i < 2 \* N; i++)

if (S[i] == 1) return 0;

return 1;

}

void InSet(char c, char S[]) // add to set

{

if (Letter(c))

if (c <= z) S[c - a] = 1; else S[c + N - A] = 1;

}

**isitmuse@isitmuse**:**~/lab/13/headers**$ cd ..

**isitmuse@isitmuse**:**~/lab/13**$ gcc -o 13 13.c

**isitmuse@isitmuse**:**~/lab/13**$ cat test1.txt

example

exam is hard, but u can pass it. exxxxxaaaaammmmm, yeeeeeeaaaaaaaahhhhhhh

**isitmuse@isitmuse**:**~/lab/13**$ ./13 < test1.txt

exxxxxaaaaammmmm

**isitmuse@isitmuse**:**~/lab/13**$ cat test2.txt

Moscow

Msc moscow woooow how? 32425.

cow mosmosmosmos hey MosMosMosMos yo

**isitmuse@isitmuse**:**~/lab/13**$ ./13 < test2.txt

MosMosMosMos

1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) 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 |
| 1 | дом | 03.11.2021 | 23:47 | Не выводится ответ  Ошибка была в функции Clear.  Условием цикла было i<=LEN | Изменить условие цикла на i<LEN |  |

1. Замечание автора по существу работы: замечания отсутствуют
2. Выводы:

В ходе данной лабораторной работы я научился работать с множествами и массивами в Си, а также я научился создавать хедер файлы.

Подпись студента