**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЕВМ**

Курсовая РАБОТА

**по дисциплине «Программирование»**

Тема: **Разработка программы на языке C для обработки текстовых данных**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3344 |  | Тукалкин В.А. |
| Преподаватель |  | Глазунов.С.А. |

Санкт-Петербург

2023

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент Тукалкин В.А. | | |
| Группа 3344 | | |
| Тема работы: "Разработка программы на языке C для обработки текстовых данных." | | |
| Исходные данные:   * Разработка должна вестись на языке программирования C. * Ввод исходных текстовых данных в стандартный поток ввода (stdin). * Вывод результатов в стандартный поток вывода (stdout). * В случае использования Makefile название исполняемого файла должно быть: cw. | | |
| Содержание пояснительной записки:   1. Содержание 2. Введение 3. Задание варианта 4. Функции 5. Полученные результаты 6. Заключение 7. Список использованных источников 8. Приложение А | | |
| Предполагаемый объем пояснительной записки:  Не менее 25 страниц. | | |
| Дата выдачи задания: 16.10.2023 | | |
| Дата сдачи реферата: 14.12.2023 | | |
| Дата защиты реферата: 14.12.2023 | | |
| Студент |  | Тукалкин В.А. |
| Преподаватель |  | Глазунов.С.А. |

**Аннотация**

Курсовая работа подразумевает создание программы на языке Си для обработки текстовых данных. Основной функционал включает в себя считывание текста, операции с предложениями и реализацию нескольких функций обработки. Программа использует динамические массивы для хранения данных. Результатом работы программы является текст, в котором удалены все повторяющиеся предложения и выполнен один из методов обработки: вывод текста, вывод текста с предложениями, которые начинаются с заглавной буквы, остальные символы строчные, удалить все предложения, в которых есть 2018, отсортировать предложения по увеличению суммы цифр встречаемых в предложении, вывести на экран все предложения, в которых встречаются все цифры хотя бы один раз.

**содержание**

|  |  |  |
| --- | --- | --- |
|  | СОДЕРЖАНИЕ | 5 |
|  | ВВЕДЕНИЕ | 6 |
| 1. | ЗАДАНИЕ ВАРИАНТА | 7 |
| 2. | ФУНКЦИИ | 10 |
| 3. | ПОЛУЧЕННЫЕ РУЗУЛЬТАТЫ | 11 |
| 4. | ПРИМЕРЫ РАБОТЫ ПРОГРАММЫ | 12 |
| 4. | ЗАКЛЮЧЕНИЕ | 15 |
| 5. | СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ | 16 |
| 6. | ПРИЛОЖЕНИЕ А. КОД ПРОГРАММЫ | 17 |

**введение**

Целью курсовой работы является разработка программы на языке программирования C для обработки текстовых данных в соответствии с определёнными условиями (см. 1. ЗАДАНИЕ ВАРИАНТА (стр.7)).

Для достижения данной цели предстоит решить следующие задачи:

1. Изучения языка программирования Си.
2. Изучение стандартных библиотек и функции в них языка Си.
3. Работа с динамическим выделением памяти.
4. Создание функций для обработки текста.
5. Обработка ошибок ввода.

**1. ЗАДАНИЕ ВАРИАНТА**

Вариант 4.2

Вывод программы должен быть произведен в стандартный поток вывода: stdout.

Ввод данных в программе в стандартный поток ввода: stdin.

В случае использования Makefile название исполняемого файла должно быть: cw.

Важно: первой строкой при запуске программы нужно выводить информацию о варианте курсовой работе и об авторе программы в строго определенном формате:

Course work for option <V>, created by <Name> <Surname>.

Где V – вариант курсовой и Имя и Фамилия, как указано в репозитории группы. Данное предложение должно быть строго первым предложением в выводе программы и является отдельной строкой (заканчивается знаком ‘\n’).

Например: Course work for option 3.2, created by Ivan Ivanov.

Ввод данных:

После вывода информацию о варианте курсовой работе программа ожидает ввода пользователем числа – номера команды:

0 – вывод текста после первичной обязательной обработки (если предусмотрена заданием данного уровня сложности)

1 – вызов функции под номером 1 из списка задания

2 – вызов функции под номером 2 из списка задания

3 – вызов функции под номером 3 из списка задания

4 – вызов функции под номером 4 из списка задания

5 – вывод справки о функциях, которые реализует программа.

Программа не должна выводить никаких строк, пока пользователь не введет число.

В случае вызова справки (опция 5) текст на вход подаваться не должен, во всех остальных случаях после выбора опции должен быть считан текст.

Признаком конца текста считается два подряд идущих символа переноса строки ‘\n’. После каждой из функций нужно вывести результат работы программы и завершить программу.

В случае ошибки и невозможности выполнить функцию по какой-либо причине, нужно вывести строку:

Error: <причина ошибки>

Каждое предложение должно выводиться в отдельной строке, пустых строк быть не должно. Текст представляет собой предложения, разделенные точкой. Предложения - набор слов, разделенные пробелом или запятой, слова - набор латинских букв, и цифр. Длина текста и каждого предложения заранее не известна.

Программа должна сохранить этот текст в динамический массив строк и оперировать далее только с ним.

Программа должна найти и удалить все повторно встречающиеся предложения (сравнивать их следует посимвольно, но без учета регистра).

Программа должна выполнить одно из введенных пользователем действий и завершить работу:

1. Преобразовать предложения так, чтобы каждое слово в нем начиналось с заглавной буквы, а остальные символы слова были строчными.
2. Удалить все предложения, в которых есть число 2018 (даже если оно внутри какого-то слова).
3. Отсортировать предложения по увеличению суммы цифр встречаемых в предложении. Если в предложении нет цифр, то сумма цифр данного предложения равняется ∞.
4. Вывести на экран все предложения, в которых встречаются все цифры хотя бы один раз.

Все сортировки должны осуществляться с использованием функции стандартной библиотеки. Использование собственных функций, при наличии аналога среди функций стандартной библиотеки, запрещается.

Все подзадачи, ввод/вывод должны быть реализованы в виде отдельной функции.

**2. ФУНКЦИИ**

* int main() – пишет "Course work for option 4.2, created by Vladimir Tukalkin.", считывает цифру и текст из консоли, обрабатывает текст, согласно условию варианта, вызывает остальные функции.
* int error\_writer(int error\_number) – выводит ошибку, которая мешает выполнению основным функциям.
* int function\_0(char\* text[], int count\_sentences) – вывод текста, после первичной обработки.
* int function\_number\_1(char\* text[], int count\_sentences) – Преобразовывает предложения так, чтобы каждое слово в нем начиналось с заглавной буквы, а остальные символы слова были строчными и выводит текст.
* int function\_number\_2(char\* text[], int count\_sentences) - Удаляет все предложения, в которых есть число 2018 (даже если оно внутри какого-то слова) и выводит текст.
* int function\_number\_3(char\* text[], int count\_sentences) - Отсортировывает предложения по увеличению суммы цифр встречаемых в предложении. Если в предложении нет цифр, то сумма цифр данного предложения равняется ∞ и выводит текст.
* int function\_number\_4(char\* text[], int count\_sentences) - Выводит на экран все предложения, в которых встречаются все цифры хотя бы один раз.
* int reference() – выводит справки о функциях, которые реализует программа.
* int comparator(const void\* x1, const void\* x2) – сравнивает строки для qsort в функции function\_number\_3.

**3. Полученные рузультаты**

Программа демонстрирует успешное выполнение поставленных задач.

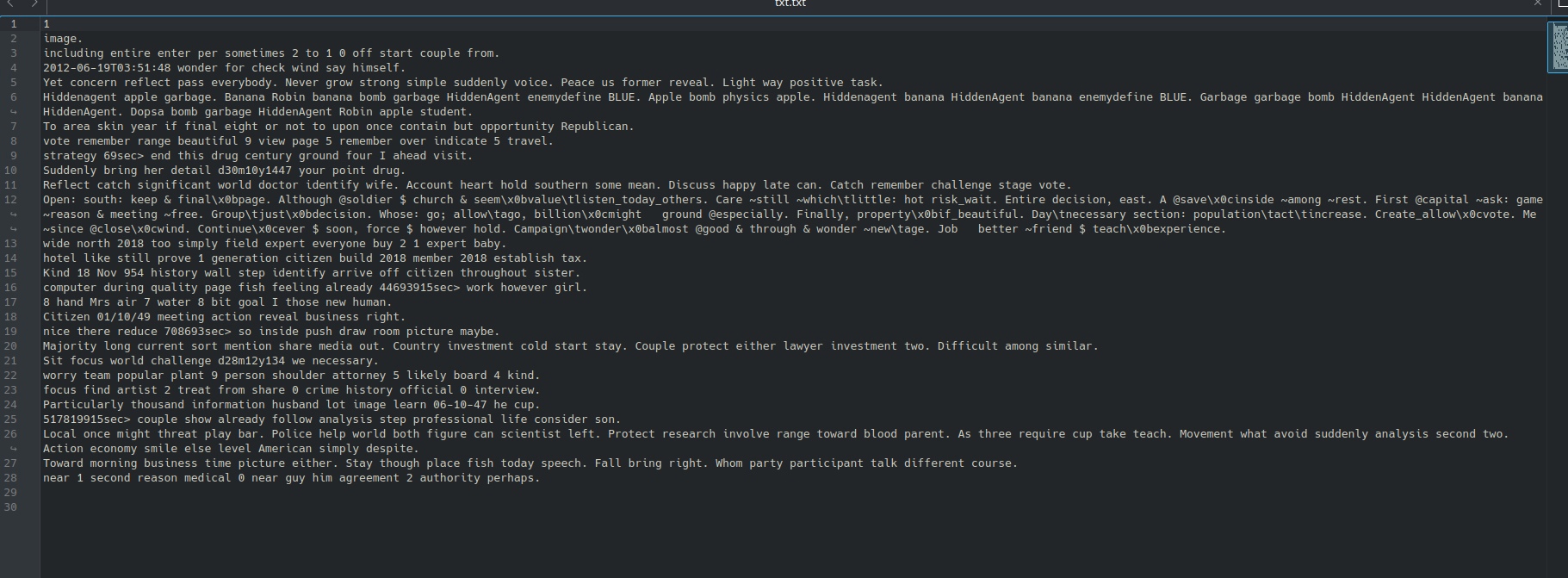
Полученные результаты включают в себя следующие ключевые аспекты:

* Программа успешно выполняет функции обработки текста, предоставляет выбор различных операций.
* Каждая функция обработки текста выполняется корректно в соответствии с поставленными требованиями.
* Реализована обработка ошибок, что позволяет программе адекватно реагировать на некорректные сценарии выполнения и оповещает какая именно ошибка.
* Программа демонстрирует стабильную работу, обеспечивая надежное выполнение операций над текстом.

**4. ПРИмеры работы программы**

Программа выполняет все функции согласно требованиям.

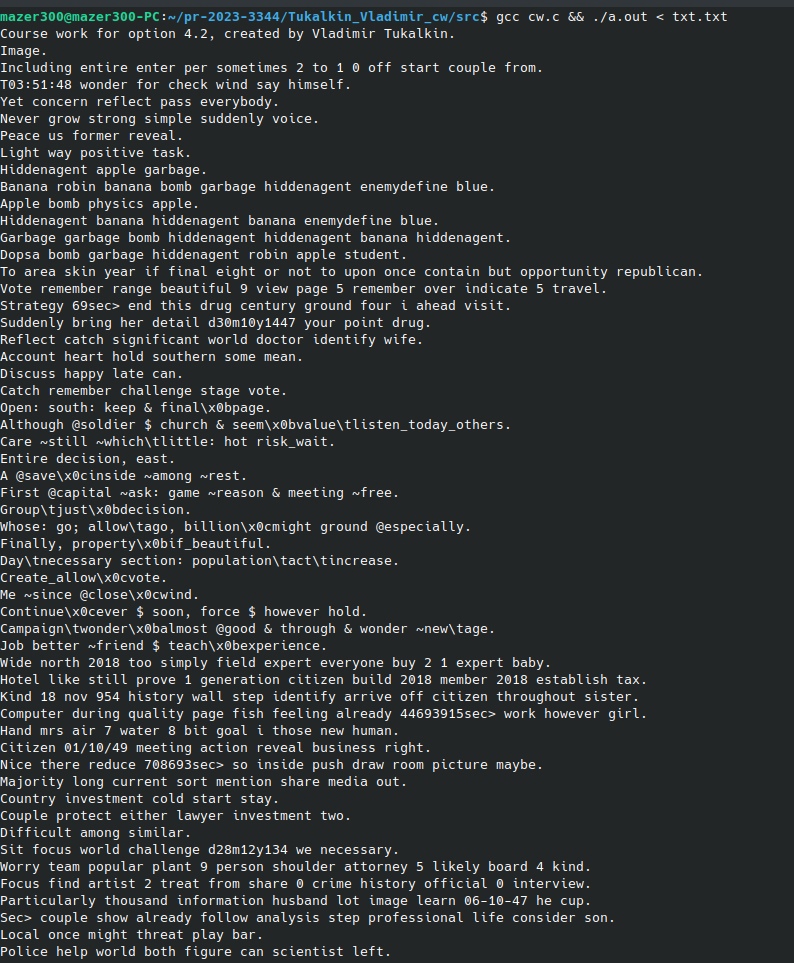
Пример текста, который использовался для теста, менялась только первая цифра:



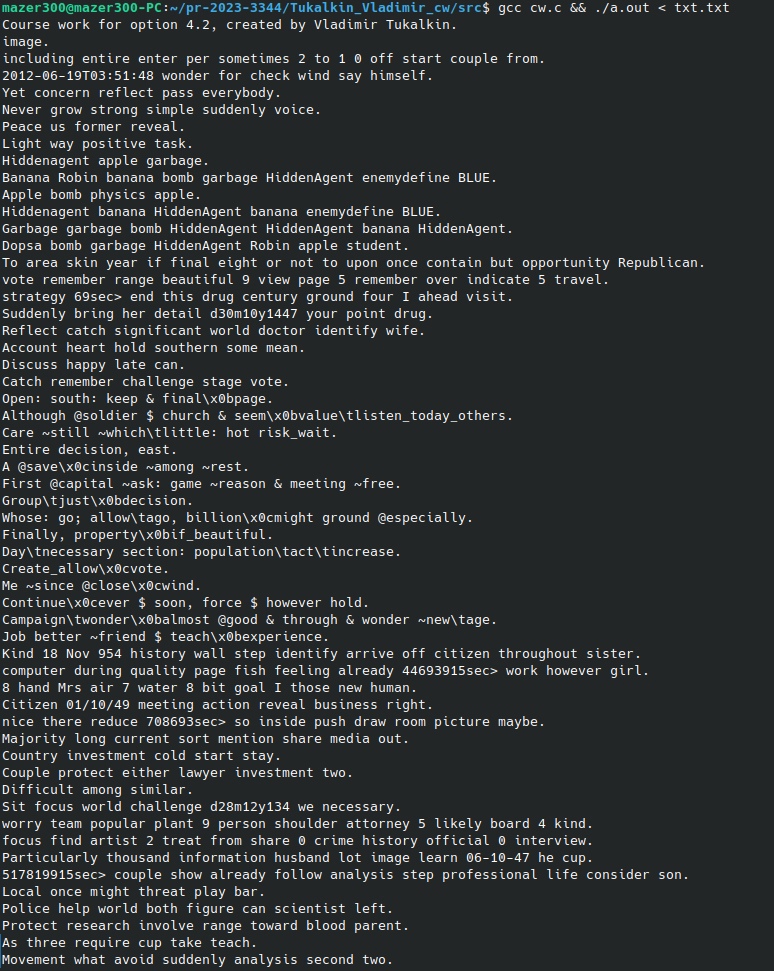
Пример для команды 0:



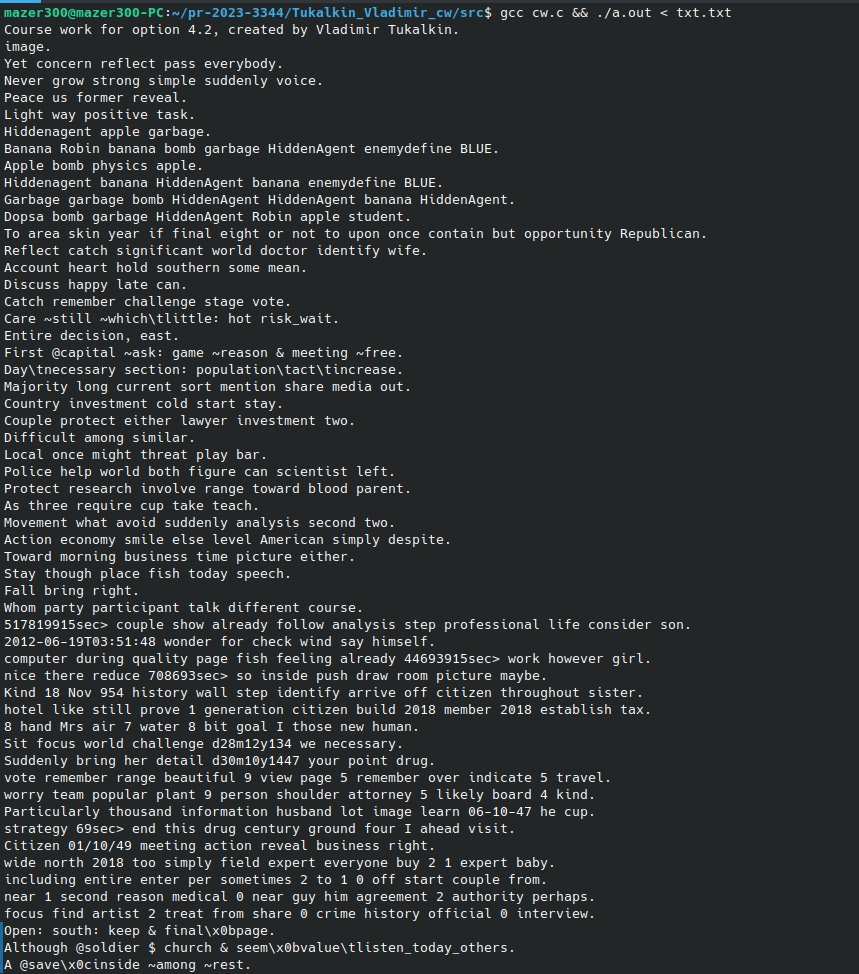
Пример для команды 1:



Пример для команды 2:



Пример для команды 3:



Пример для команды 4:



**заключение**

Программа успешно реализована и успешно выполняет поставленные задачи. В ходе разработки были задействованы стандартные библиотеки языка Си и динамическое выделение памяти. При выполнении работы были получены навыки работы с языком Си.

**список использованных источников**

1. Керниган Б.В. ЯЗЫК С.: Вильямс, 2019. 288 с.

2. Основные библиотеки в Си // Основные библиотеки в Си. URL: <https://1ww1s.github.io/Code.c/learning/libraries.html> (дата обращения: 08.12.2023).

**приложение А**

**Код программы**

Название файла: cw.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define RED "\033[1;31m"

#define RESET "\033[0m"

void error\_writer(int error\_number);

void function\_0(char\* text[], int count\_sentences);

void function\_number\_1(char\* text[], int count\_sentences);

void function\_number\_2(char\* text[], int count\_sentences);

void function\_number\_3(char\* text[], int count\_sentences);

void function\_number\_4(char\* text[], int count\_sentences);

void reference();

int main(){

printf("Course work for option 4.2, created by Vladimir Tukalkin.\n");

char command\_number;

scanf("%s",&command\_number);

if(strchr("012345",command\_number)==NULL){

error\_writer(0);

}

if(command\_number=='5') reference();

if(strchr("01234",command\_number)){

char \*\*text=malloc(sizeof(char\*));

int count\_sentences=0;

int flag\_mark=0; //check "!" and "?"

while(1){ //start of text reading

char \*sentence=malloc(sizeof(char));

char symbol=1;

int len\_sentence=0;

int count\_n=0; //check EOF

int flag\_space=0; //check extra space

int flag\_dot=0; //check space before dot

while(strchr(".!?",(int)symbol)==NULL){

symbol=getchar(); //reading

if(strchr("?!",(int)symbol)!=NULL){ //check "!" and "?"

error\_writer(1); //

flag\_mark=1; //

break; //

} //check "!" and "?"

if(symbol=='\n'){ //check \n

count\_n++; //

if(count\_n==2){ //

symbol='\n'; //

}else{ //

symbol=' '; //

} //

}else{ //

count\_n=0; //

} //check \n

if(strchr(" \t",symbol)!=NULL){ //check extra space

flag\_space++; //

symbol=' '; //

}else{ //

flag\_space=0; //

} //

if(flag\_space>1) continue; //check extra space

if(symbol==' '){ //check space before dot

flag\_dot++; //

}if(strchr("., ",symbol)==NULL){ //

flag\_dot=0; //

} //

if((symbol=='.' || symbol==',') && flag\_dot>0){ //

sentence[len\_sentence-1]=symbol; //

flag\_dot=0; //

if(symbol==','){ //

sentence[len\_sentence++]=' '; //

sentence=realloc(sentence,sizeof(char)\*(len\_sentence+1)); //

} //

flag\_space++; //

continue; //

} //check space before dot

if(strchr(".,",symbol)!=NULL){ //space after comma

sentence[len\_sentence++]=symbol; //

sentence=realloc(sentence,sizeof(char)\*(len\_sentence+1)); //

if(symbol==','){ //

sentence[len\_sentence++]=' '; //

sentence=realloc(sentence,sizeof(char)\*(len\_sentence+1)); //

flag\_space++; //

} //

continue; //

} //space after comma

if(flag\_space<2){

sentence[len\_sentence++]=symbol;

sentence=realloc(sentence,sizeof(char)\*(len\_sentence+1));

}

if(count\_n==2){ //EOF

symbol='.'; //

sentence[len\_sentence++]=symbol; //

sentence=realloc(sentence,sizeof(char)\*(len\_sentence)); //

} //EOF

}

if(flag\_mark==1) break;

if(strstr(sentence,"\n.")) break; //EOF

sentence[len\_sentence++]='\n';

sentence=realloc(sentence,sizeof(char)\*(len\_sentence+1));

sentence[len\_sentence]='\0';

if(strchr(" ",sentence[0])) memmove(sentence,sentence+1,strlen(sentence));

if(strlen(sentence)!=1 && strchr(" \t",sentence[0])==NULL){ //writing sentence in text

text[count\_sentences++]=sentence; //

text=realloc(text,sizeof(char\*)\*(count\_sentences+1)); //

} //writing sentence in text

} //end of text reading

for(int number\_str1=0;number\_str1<count\_sentences-1;number\_str1++){ //start of compare sentence

for(int number\_str2=1;number\_str2<count\_sentences;number\_str2++){

if(strlen(text[number\_str1])==strlen(text[number\_str2]) && number\_str1!=number\_str2){

int count\_sameness=0;

for(int i=0;i<strlen(text[number\_str1]);i++){

if(tolower(text[number\_str1][i])==tolower(text[number\_str2][i])){

count\_sameness++;

}else{

count\_sameness=0;

break;

}

if(count\_sameness==strlen(text[number\_str1])){

char\* str=malloc(sizeof(char));

str[0]=' ';

text[number\_str2]=realloc(text[number\_str2],sizeof(char));

text[number\_str2]=str;

}

}

}

}

} //end of compare sentence

if(flag\_mark==0){

switch (command\_number){

case '0':

function\_0(text, count\_sentences);

break;

case '1':

function\_number\_1(text, count\_sentences);

break;

case '2':

function\_number\_2(text, count\_sentences);

break;

case '3':

function\_number\_3(text, count\_sentences);

break;

case '4':

function\_number\_4(text, count\_sentences);

break;

}

}

for(int i=0;i<count\_sentences;i++) free(text[i]); //clean memory

free(text);

}

}

void error\_writer(int error\_number){ //writing error

switch (error\_number){

case 0:

printf("\n%sError: invalid command number%s\n",RED,RESET);

break;

case 1:

printf("\n%sError: '!' or '?' at the end of the sentence%s\n",RED,RESET);

break;

}

}

void function\_0(char\* text[], int count\_sentences){

for(int i=0;i<count\_sentences;i++){

if(strlen(text[i])>2) printf("%s",text[i]);

}

}

void function\_number\_1(char\* text[], int count\_sentences){

for(int i=0;i<count\_sentences;i++){

if(strlen(text[i])>2){

int flag\_start=0;

int count\_num=0;

for(int j=0;j<strlen(text[i]);j++){

char\* symbol=malloc(sizeof(char));

strncpy(symbol,text[i]+j,1);

if(strchr("qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM",\*symbol)==NULL) count\_num++;//writing of the first capital letter and other sentence

if(strchr("qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM",\*symbol)==NULL && flag\_start==0){

continue;

}else{

if(j-count\_num==0){

printf("%c",toupper(\*symbol));

}else{

printf("%c",tolower(\*symbol));

}

flag\_start=1;

free(symbol);

}

}

}

}

}

void function\_number\_2(char\* text[], int count\_sentences){

for(int i=0;i<count\_sentences;i++){

if(strstr(text[i],"2018")==NULL) printf("%s",text[i]);

}

}

int comparator(const void\* x1, const void\* x2){

const char \*\*str1=(const char \*\*)x1;

const char \*\*str2=(const char \*\*)x2;

int count1=0, count2=0;

if(strstr(\*str1,"0")==NULL && strstr(\*str1,"1")==NULL && strstr(\*str1,"2")==NULL && strstr(\*str1,"3")==NULL && strstr(\*str1,"4")==NULL && strstr(\*str1,"5")==NULL && strstr(\*str1,"6")==NULL && strstr(\*str1,"7")==NULL && strstr(\*str1,"8")==NULL && strstr(\*str1,"9")==NULL) return -1;

if(strstr(\*str2,"0")==NULL && strstr(\*str2,"1")==NULL && strstr(\*str2,"2")==NULL && strstr(\*str2,"3")==NULL && strstr(\*str2,"4")==NULL && strstr(\*str2,"5")==NULL && strstr(\*str2,"6")==NULL && strstr(\*str2,"7")==NULL && strstr(\*str2,"8")==NULL && strstr(\*str2,"9")==NULL) return 1;

else{

for(int i=0;i<strlen(\*str1);i++){

char symbol;

strncpy(&symbol,&(\*str1)[i],1);

if(strchr("1234567890",symbol)!=NULL) count1=count1+atoi(&symbol);

}

for(int i=0;i<strlen(\*str2);i++){

char symbol;

strncpy(&symbol,&(\*str2)[i],1);

if(strchr("1234567890",symbol)!=NULL) count2=count2+atoi(&symbol);

}

return count2-count1;

}

}

void function\_number\_3(char\* text[], int count\_sentences){

qsort(text,count\_sentences,sizeof(char\*),comparator);

for(int i=0;i<count\_sentences;i++) printf("%s",text[i]);

}

void function\_number\_4(char\* text[], int count\_sentences){

for(int i=0;i<count\_sentences;i++){

if(strstr(text[i],"0")!=NULL && strstr(text[i],"1")!=NULL && strstr(text[i],"2")!=NULL && strstr(text[i],"3")!=NULL && strstr(text[i],"4")!=NULL && strstr(text[i],"5")!=NULL && strstr(text[i],"6")!=NULL && strstr(text[i],"7")!=NULL && strstr(text[i],"8")!=NULL && strstr(text[i],"9")!=NULL) printf("%s",text[i]);

}

}

void reference(){

printf("Reference:\n0-text output after the initial mandatory processing\n1-convert sentences so that each word in it starts with a capital letter, and the remaining characters of the word are lowercase\n2-Delete all offers that contain the number 2018\n3-sort the sentences by increasing the sum of the digits found in the sentence. If there are no digits in the sentence, then the sum of the digits of this sentence is infinity\n4-display all sentences in which all numbers occur at least once\n");

}