МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

КУРСОВАЯ РАБОТА

по дисциплине «Программирование»

Тема: Работа со строками в языке С.

Студент гр. 1304 _____ Климов Г. А.

Преподаватель _____ Чайка К. В.

Санкт-Петербург

2021

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студент Климов Г. А.

Группа 1304

Тема работы : работа со строками в языке С.

Исходные данные:

Вводится текст, состоящий из предложений разделённых точкой.

Содержание пояснительной записки:

«Содержание», «Введение», «Заключение», «Список использованных источников»

Дата выдачи задания: 15.10.2021

Дата сдачи реферата: 26.12.2021

Дата защиты реферата: 27-30.12.2021

Студент _____ Климов Г. А.

Преподаватель _____ Чайка К. В.

АННОТАЦИЯ

Работа представляет из себя программу, предполагающую работу с текстом. Программа должна сохранить (считать) текст в виде динамического массива предложений и оперировать далее только с ним. Для хранения предложения и для хранения текста требуется реализовать структуры Sentence и Text. Функции обработки также должны принимать на вход либо текст (Text), либо предложение (Sentence).

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель работы — написание программы для считывания и редактирования строк.

Задание: программа должна сохранить (считать) текст в виде динамического массива предложений, удалить одинаковые предложения и реализовать следующие функции:

1) Для каждого предложения вывести строку-дату вида “ДД-ММ-ГГГГ”, где день - количество слов в предложении, месяц - наибольшая длина слова в предложении, год - общее количество символов в предложении + 1900.

2) Вывести предложения так, чтобы слова шли в обратном порядке.

3) Отсортировать предложения по длине первого слова в предложении.

4) Удалить все предложения у которых все слова имеют длину не больше 3 символов.

1. ОПИСАНИЕ ПРОГРАММЫ

1.1. Основая функция

```
#include "func.h"

int main(){
	char c;
	int g;
	int i;
	setlocale(LC_ALL,"");
	wprintf(L"Ведите текст\n");
	struct Text tex =read();
	int n = tex.size;
	tex.t=delete_copy(tex);
	choice();
	c=getwchar();
	g=c -'0';

	switch (g) {
	case 1:data(tex);
			break;
	case 2:reverse(tex);
			break;
	case 3:sort(tex);
			break;
	case 4:delete(tex);
			break;
	case 5:
			wprintf(L"Программа остановлена\n");
```

```
    }

    for(i=0;i<n;i++){
        free(tex.t[i]);
    }
    free(tex.t);

 return 0;
}
```

Функция предлагает пользователю ввести текст. Далее она сохраняет его в виде динамического массива благодоря функции read(). И удаляют копии предложений с помощью фунции delete_copy. Позже она даёт пользователю выбор действий и вызывает функции обработки текста.В конце очищает память.

1.2. Функция удаления одинаковых предложений

```
wchar_t ** delete_copy(struct Text parm)
{
    int n =parm.size;
    setlocale(LC_ALL,"");
    int q;
    int i;
    int j;
    int y;
    for(i =0;i < n;i++){
        for ( y = 0; y < wcslen(parm.t[i]) ; y++) {
        parm.t[i][y] = towlower(parm.t[i][y]);


    }
    }
```

```
 i=0;
while(i<n-1){
j=i+1;
while(j<n){
        q=0;
        if (wcslen(parm.t[i])== wcslen(parm.t[j])){
        for (int y = 0; y< wcslen(parm.t[i]) ; y++) {
        if (parm.t[i][y] ==parm.t[j][y]){
                        q=q+1;
                        }
                }
                if(q==wcslen(parm.t[i])){
                        for (int y = 0; y< wcslen(parm.t[j]) ; y++){
                        parm.t[j][y]=' ';
                        }
                }
        }
j++;
}
i++;
}

return(parm.t);
}
```

Так как сравнение должно быть посимвольным и без учёта регистра, то функция меняет регистр всех предложений, а дальше сравнивает их.

1.3 Функция вывода строки в формате даты

void data(struct Text parm){

```
int n =parm.size;
setlocale(LC_ALL,"");
for(int i=0;i<n-1;i++){
        int slovo =1;
        int maxsl=0;
        int max=0;
        int q=0;
        int k=0;;

        if(parm.t[i][0] != ' '){
        for (int y = 0; y < wcslen(parm.t[i]); y++) {
                if (parm.t[i][y] == ' '){
                slovo=slovo+1;
                if(max<maxsl){
                max=maxsl;
                maxsl=0;
                q=1;
                }
                maxsl =0;
                }else{
                if (parm.t[i][y] != ','){
                maxsl=maxsl+1;
                }
                }
      }
        int d=wcslen(parm.t[i]) + 1900 -1;
        if(q==1 && max<maxsl){
        max=maxsl-1;
        }
```

```
        if(q==0){
        max=maxsl-1;
        }

       wprintf(L"%d%c%d%c%d \n",slovo,'-',max,'-',d);
   }
}

}
```

Функция работает с предложением, определяя количество слов, максимальную длину слова, потом выводит нужный результат.

1.4. Функция вывода предложений в обратном порядке

```
void reverse(struct Text parm){
        int n =parm.size;
        setlocale(LC_ALL,"");
        for(int i=0;i<n-1;i++){
        if (parm.t[i][0] != ' '){
        int len =wcslen(parm.t[i]);
        for (int y = len-1; y >= 0; y--) {
                putwchar(parm.t[i][y]);
        }
        wprintf(L"  \n");
        }

 }

 }
```

Функция циклом for посимвольно выводит предложение наоборот.

1.5. Функция вывода предложений по длине первого слова

```
void sort(struct Text parm){
```

```
        int q=0;
        int n =parm.size;
        setlocale(LC_ALL,"");
        int tmp;
        int arr[n];
        int arr2[n];
        for(int i=0;i<n;i++){
        int maxsl=0;
        if(parm.t[i][0] != ' ' ){
        for (int y = 0; y < wcslen(parm.t[i]); y++) {
                if (parm.t[i][y] == ' '|| parm.t[i][y]=='.' ){
                break;
                }else{
                if (parm.t[i][y] != ','){
                maxsl=maxsl+1;
                }
                }
        }
        arr[i]=maxsl;
        arr2[i]=maxsl;
  }

}
for(int i=0; i<n; i++){
          for(int j=i+1; j<n; j++){
             if (arr2[i] < arr2[j])
             {
                tmp = arr2[j];
                arr2[j] = arr2[i];
```

```
            arr2[i] = tmp;
        }
    }
}
for(int i=0;i<n-1;i++){

    for(int j=0; j<n-1; j++){
        if(arr2[i]==arr[j] && arr2[i]!=0){
        arr[j]=-1;
        wprintf(L"%ls \n",parm.t[j]);

        }

        }
    }


}
```

Функция находит первые слова предложений, далее создаются массивы в которые записываются длины этих слов. Один из массивов сортируется и сравниваются значения отсортированного массива и массива с длиной слова у соотвествующего предложения. После предложения выводятся на экран.

1.6. Функция удаления всех предложений со словами короче 3 символов

```
    void delete(struct Text parm){
    int n =parm.size;
    setlocale(LC_ALL,"");
    for(int i=0;i<n-1;i++){
        int q=0;
        int maxsl=0;
```

```
        int max=0;
        if(parm.t[i][0] != ' ' ){
        for (int y = 0; y < wcslen(parm.t[i]); y++) {
                if (parm.t[i][y] == ' '){
                if(max<maxsl){
                max=maxsl;
                maxsl=0;
                q=1;
                }
                maxsl =0;
                }else{
                if (parm.t[i][y] != ','){
                maxsl=maxsl+1;
                }
                }
      }

if(q==1 && max<maxsl){
        max=maxsl-1;
        }
        if(q==0){
        max=maxsl-1;
        }
 if(max>3){
        wprintf(L"%ls\n",parm.t[i]);
        }
}
}
```

}

Функция находит максимальную длину слова в предложении, если она меньше трёх, то предложение удаляется.

1.7 Функция ввода текста.

```
struct Text read(){
        struct Text tex;
        struct Sentence s;
        int m,n,counter,i,d,q;
        d=1;
        wchar_t c;
        tex.t=malloc(sizeof(wchar_t*));
        n=0;
        m=0;
        while (d == 1){
                tex.t=realloc(tex.t,sizeof(wchar_t*)*(n+2));
                tex.t[n]=malloc(sizeof(wchar_t)*2);
                c=getwchar();
                if (c == '\n'){
                        d=0;
                } else {
                tex.t[n][0]=c;
                tex.t[n][1]='\0';
                counter=0;
                q=0;
                while ((c!='.') ){
                        counter= counter +1;
                        tex.t[n]=realloc(tex.t[n],sizeof(wchar_t)*(counter+2));
                        c=getwchar();
```

```
                tex.t[n][counter]=c;
                tex.t[n][counter+1]='\0';
            }
            }
            s.s=tex.t[n];
            n=n+1;

        }
    tex.size =n;
    return tex;

    }
```

Функция сохраняет текст в виде динамического массива предложений в структуре.

1.8 Функция вывода действий пользователя

```
void choice(){
fputws(L"1 - Для каждого предложения вывести строку-дату вида “ДД-ММ-ГГГГ”, где день - количество слов в предложении, месяц - наибольшая длина слова в предложении, год - общее количество символов в предложении + 1900.\n",stdout);
        fputws(L"2 - Вывести предложения так, чтобы слова шли в обратном порядке.\n",stdout);
        fputws(L"3 - Отсортировать предложения по длине первого слова в предложении.\n",stdout);
        fputws(L"4 - Удалить все предложения у которых все слова имеют длину не больше 3 символов.\n",stdout);
        fputws(L"5 - Выход из программы.\n",stdout);
```

}

Функция выводит действия которые может совершить пользователь

2. СБОРКА ПРОГРАММЫ

Так как надо было создать Makefile к данной программе, то я реализовал данную задачу так.
Все функции обработки текста(все кроме main) были сгруппированы в файле func.c, кроме функций read() и choice(), так как они являются функциями ввода и вывода текста, они сгруппированы ы вайле str.c.Все функции объявлены в файле func.h.
главная же функция в файле main.c.
Сборка осуществляется утилитой make в соотвествии с инструкцией в Makefile.

3.ТЕСТИРОВАНИЕ

ввод	вывод	комментарии
Жито как жито.Чё, кто это.Це я, кто ж ещё. 4	жито как жито	Задание выполнено корректно
а роза упала на лапу азора. 2	.ароза упал ан алапу азор а	Задание выполнено корректно
тук тук.кто там.тук тук.это я.что значит я.я.кто ж ещё. 3	тук тук. кто там. это я. что значит я. кто ж ещё. я.	Задание выполнено корректно

4.ПОЛЬЗОВАТЕЛЬСКАЯ ИНСТРУКЦИЯ

Откройте в терминале папку с файлами данной программы.Далее введите слово make.После чего вводите ./main и программа запускается.Далее вводите текст и выбираете одно из доступных действий.

ЗАКЛЮЧЕНИЕ

Были изучены теоретические материалы по теме курсовой работы, разработан и написан программный код, проведено тестирование программы.

Исходный код в приложении А.

Результаты тестов в приложении Б.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. All-ht.ru [Электроный ресурс]

2.Prog-cpp.ru [Электроный ресурс]

3. habr.ru [Электроный ресурс]

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

файл main.c

```
#include "func.h"

int main(){
	char c;
	int g;
	int i;
	setlocale(LC_ALL,"");
	wprintf(L"Ведите текст\n");
	struct Text tex =read();
	int n = tex.size;
	tex.t=delete_copy(tex);
	choice();
	c=getwchar();
	g=c -'0';

	switch (g) {
	case 1:data(tex);
			break;
	case 2:reverse(tex);
		break;
	case 3:sort(tex);
			break;
	case 4:delete(tex);
			break;
	case 5:
			wprintf(L"Программа остановлена\n");
	}

	for(i=0;i<n;i++){
		free(tex.t[i]);
	}
	free(tex.t);

 return 0;
}
```

файл func.c

```
#include "func.h"



wchar_t ** delete_copy(struct Text parm)
```

```
{
	int n =parm.size;
	setlocale(LC_ALL,"");
	int q;
	int i;
	int j;
	int y;
	for(i =0;i < n;i++){
		for ( y = 0; y < wcslen(parm.t[i]) ; y++) {
		parm.t[i][y] = towlower(parm.t[i][y]);


	}
	}

	 i=0;
	while(i<n-1){
	j=i+1;
	while(j<n){
		q=0;
		if (wcslen(parm.t[i])== wcslen(parm.t[j])){
		for (int y = 0; y< wcslen(parm.t[i]) ; y++) {
		if (parm.t[i][y] ==parm.t[j][y]){
				q=q+1;
				}
			}
			if(q==wcslen(parm.t[i])){
				for (int y = 0; y< wcslen(parm.t[j]) ; y++){
				parm.t[j][y]=' ';
				}
			}
		}
	j++;
	}
	i++;
	}

return(parm.t);
}


void data(struct Text parm){
int n =parm.size;
setlocale(LC_ALL,"");
for(int i=0;i<n-1;i++){
	int slovo =1;
	int maxsl=0;
	int max=0;
	int q=0;
	int k=0;;
```

```
        if(parm.t[i][0] != ' ' ){
        for (int y = 0; y < wcslen(parm.t[i]); y++) {
                if (parm.t[i][y] == ' '){
                slovo=slovo+1;
                if(max<maxsl){
                max=maxsl;
                maxsl=0;
                q=1;
                }
                maxsl =0;
                }else{
                if (parm.t[i][y] != ','){
                maxsl=maxsl+1;
                }
                }
      }
        int d=wcslen(parm.t[i]) + 1900 -1;
        if(q==1 && max<maxsl){
        max=maxsl-1;
        }
        if(q==0){
        max=maxsl-1;
        }

       wprintf(L"%d%c%d%c%d \n",slovo,'-',max,'-',d);
   }
}

}

void delete(struct Text parm){
int n =parm.size;
setlocale(LC_ALL,"");
for(int i=0;i<n-1;i++){
        int q=0;
        int maxsl=0;
        int max=0;
        if(parm.t[i][0] != ' ' ){
        for (int y = 0; y < wcslen(parm.t[i]); y++) {
                if (parm.t[i][y] == ' '){
                if(max<maxsl){
                max=maxsl;
                maxsl=0;
                q=1;
                }
                maxsl =0;
                }else{
                if (parm.t[i][y] != ','){
                maxsl=maxsl+1;
                }
                }
```

```
      }

if(q==1 && max<maxsl){
          max=maxsl-1;
          }
          if(q==0){
          max=maxsl-1;
          }
 if(max>3){
          wprintf(L"%ls\n",parm.t[i]);
          }
}
}


}

void reverse(struct Text parm){
          int n =parm.size;
          setlocale(LC_ALL,"");
          for(int i=0;i<n-1;i++){
          if (parm.t[i][0] != ' ' ){
          int len =wcslen(parm.t[i]);
          for (int y = len-1; y >= 0; y--) {
                    putwchar(parm.t[i][y]);
          }
          wprintf(L"  \n");
          }

 }

 }

void sort(struct Text parm){
          int q=0;
          int n =parm.size;
          setlocale(LC_ALL,"");
          int tmp;
          int arr[n];
          int arr2[n];
          for(int i=0;i<n;i++){
          int maxsl=0;
          if(parm.t[i][0] != ' ' ){
          for (int y = 0; y < wcslen(parm.t[i]); y++) {
                    if (parm.t[i][y] == ' '|| parm.t[i][y]=='.' ){
                    break;
                    }else{
                    if (parm.t[i][y] != ','){
                    maxsl=maxsl+1;
                    }
                    }
```

```
        }
        arr[i]=maxsl;
        arr2[i]=maxsl;
  }

}
for(int i=0; i<n; i++){
        for(int j=i+1; j<n; j++){
           if (arr2[i] < arr2[j])
           {
              tmp = arr2[j];
              arr2[j] = arr2[i];
              arr2[i] = tmp;
           }
        }
}
for(int i=0;i<n-1;i++){

         for(int j=0; j<n-1; j++){
                if(arr2[i]==arr[j] && arr2[i]!=0){
                arr[j]=-1;
                wprintf(L"%ls \n",parm.t[j]);

                }

                }
         }


}
```

файл func.h

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <wchar.h>
#include <locale.h>
#include <wctype.h>
```

```
struct Text
{
wchar_t **t;
int size;
};
struct Sentence{
wchar_t *s;
};
struct Text read();
void choice();
wchar_t ** delete_copy(struct Text parm);
void data(struct Text parm);
void delete(struct Text parm);
void reverse(struct Text parm);
void sort(struct Text parm);
```

файл str.c

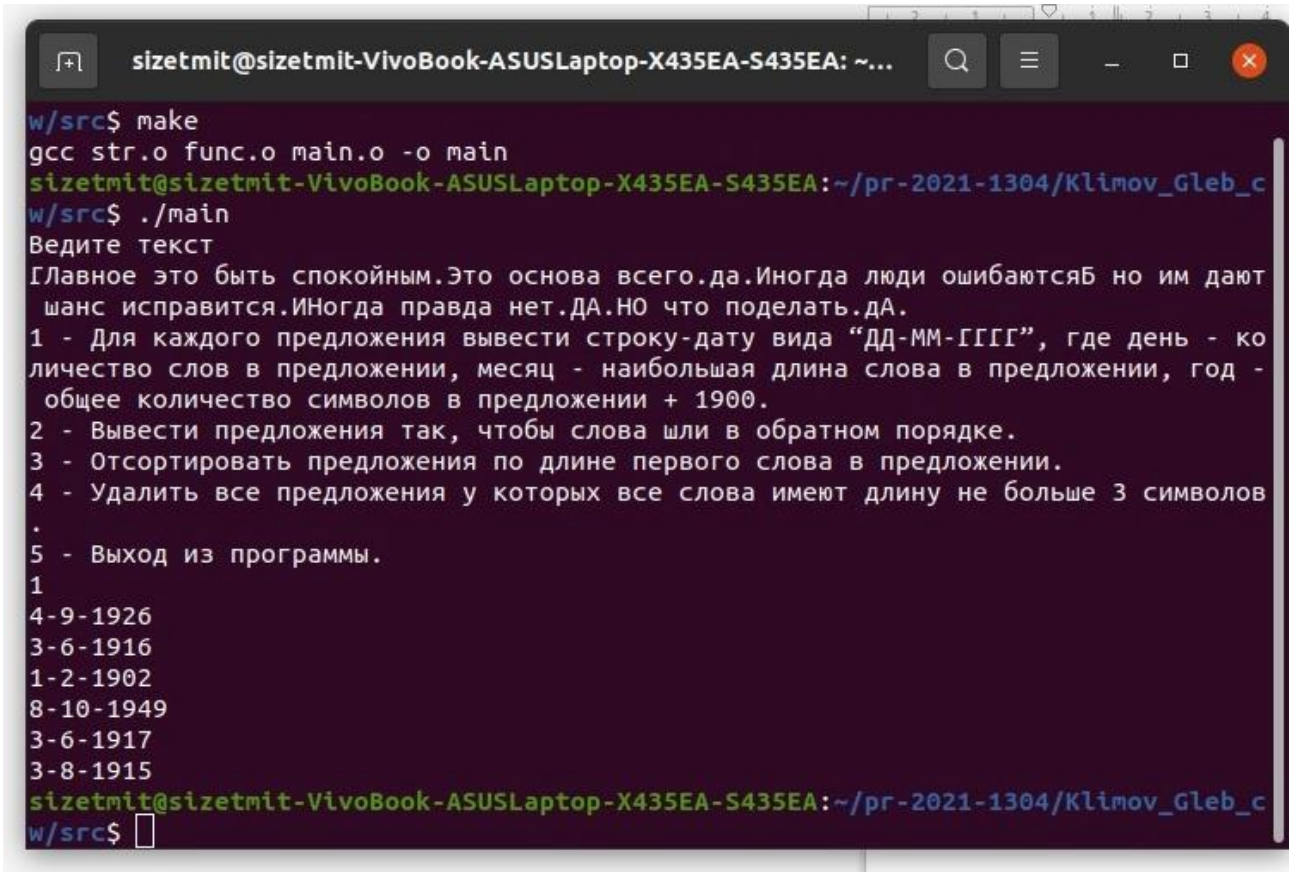
```
#include "func.h"

struct Text read(){
        struct Text tex;
        struct Sentence s;
        int m,n,counter,i,d,q;
        d=1;
        wchar_t c;
        tex.t=malloc(sizeof(wchar_t*));
        n=0;
        m=0;
        while (d == 1){
                tex.t=realloc(tex.t,sizeof(wchar_t*)*(n+2));
                tex.t[n]=malloc(sizeof(wchar_t)*2);
                c=getwchar();
                if (c == '\n'){
                        d=0;
                } else {
                tex.t[n][0]=c;
                tex.t[n][1]='\0';
                counter=0;
                q=0;
                while ((c!='.') ){
                        counter= counter +1;
                        tex.t[n]=realloc(tex.t[n],sizeof(wchar_t)*(counter+2));
                        c=getwchar();
                        tex.t[n][counter]=c;
                        tex.t[n][counter+1]='\0';
                }
                }
                s.s=tex.t[n];
                n=n+1;

        }
```

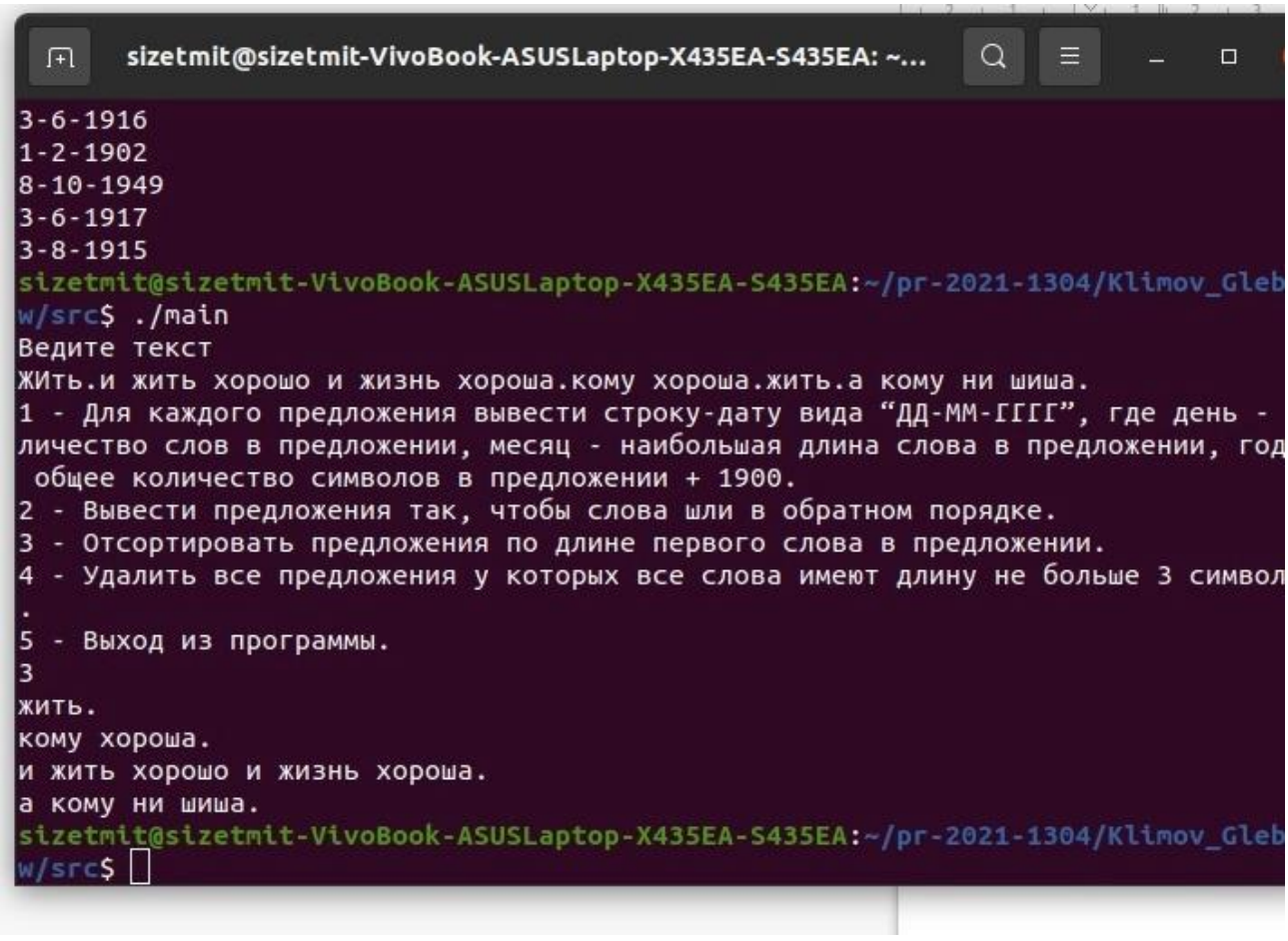
```
tex.size =n;
return tex;

}


void choice(){
fputws(L"1 - Для каждого предложения вывести строку-дату вида “ДД-ММ-ГГГГ”, где день - количество слов в предложении, месяц - наибольшая длина слова в предложении, год - общее количество символов в предложении + 1900.\n",stdout);
        fputws(L"2 - Вывести предложения так, чтобы слова шли в обратном порядке.\n",stdout);
        fputws(L"3 - Отсортировать предложения по длине первого слова в предложении.\n",stdout);
        fputws(L"4 - Удалить все предложения у которых все слова имеют длину не больше 3 символов.\n",stdout);
        fputws(L"5 - Выход из программы.\n",stdout);

}
```

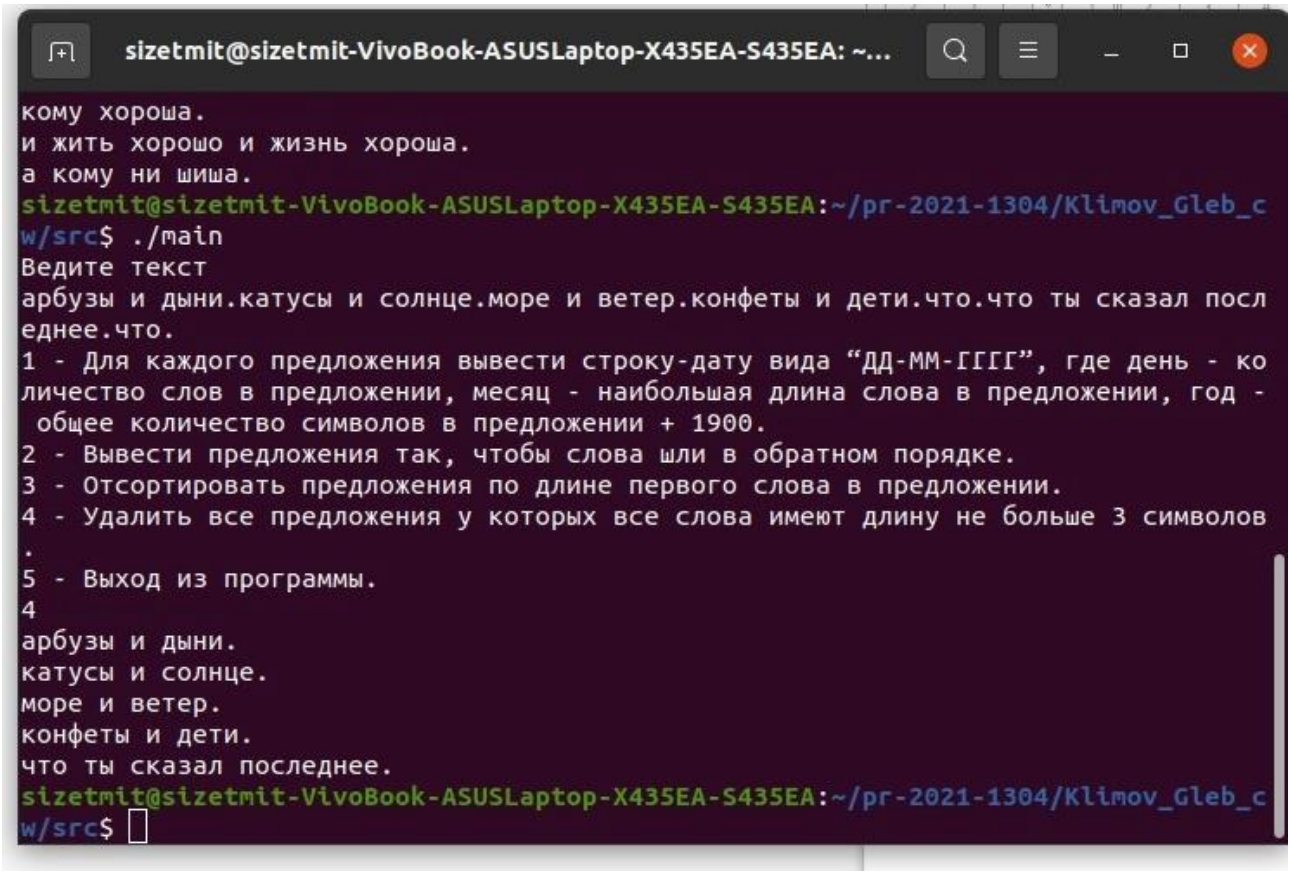
Makefile

```
all: str.o func.o main.o
        gcc str.o func.o main.o -o main
str.o: str.c func.h
        gcc -c str.c
func.o: func.c func.h
        gcc -c func.c
main.o: main.c func.h
        gcc -c main.c
```

ПРИЛОЖЕНИЕ Б
СКРИНШОТЫ ЗАПУСКА ПРОГРАММЫ






sizetmit@sizetmit-VivoBook-ASUSLaptop-X435EA-S435EA: ~...
кому хороша.
и жить хорошо и жизнь хороша.
а кому ни шиша.
sizetmit@sizetmit-VivoBook-ASUSLaptop-X435EA-S435EA:~/pr-2021-1304/Klimov_Gleb_c
w/src$ ./main
Ведите текст
арбузы и дыни.катусы и солнце.море и ветер.конфеты и дети.что.что ты сказал посл
еднее.что.
1 - Для каждого предложения вывести строку-дату вида “ДД-ММ-ГГГГ”, где день - ко
личество слов в предложении, месяц - наибольшая длина слова в предложении, год -
общее количество символов в предложении + 1900.
2 - Вывести предложения так, чтобы слова шли в обратном порядке.
3 - Отсортировать предложения по длине первого слова в предложении.
4 - Удалить все предложения у которых все слова имеют длину не больше 3 символов
.
5 - Выход из программы.
4
арбузы и дыни.
катусы и солнце.
море и ветер.
конфеты и дети.
что ты сказал последнее.
sizetmit@sizetmit-VivoBook-ASUSLaptop-X435EA-S435EA:~/pr-2021-1304/Klimov_Gleb_c
w/src$