Национальный исследовательский ядерный университет «МИФИ»

Институт интеллектуальных кибернетических систем

Кафедра №12 «Компьютерные системы и технологии»

**ОТЧЕТ**

**О выполнении лабораторной работы №4**

**«Работа со строками»**

**Cтудент**: Гатченко А.С.

**Группа**: Б22-525

**Преподаватель**: Половнева Ю. А.

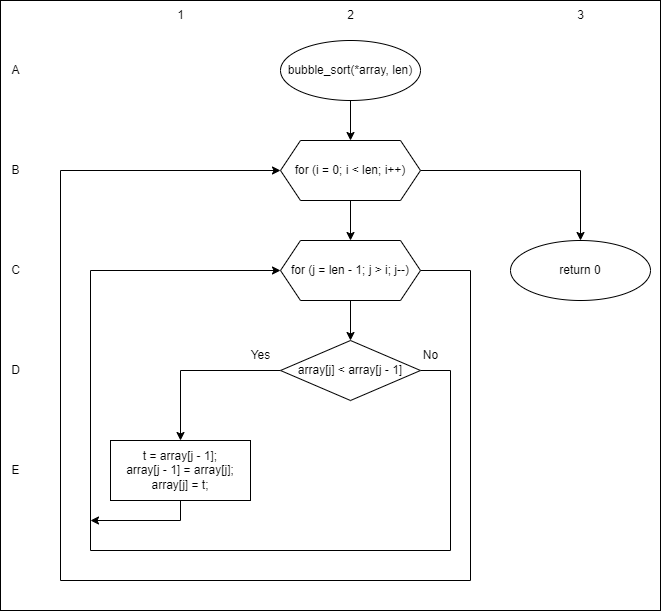
**1. Формулировка индивидуального задания**

Упорядочить слова в строке по возрастанию длины.

**2. Описание использованных типов данных**

При выполнении данной лабораторной работы использовались встроенные типы данных int, предназначенные для работы с целыми числами, а также указатели на символы и на массивы символов.

**3. Описание использованного алгоритма**



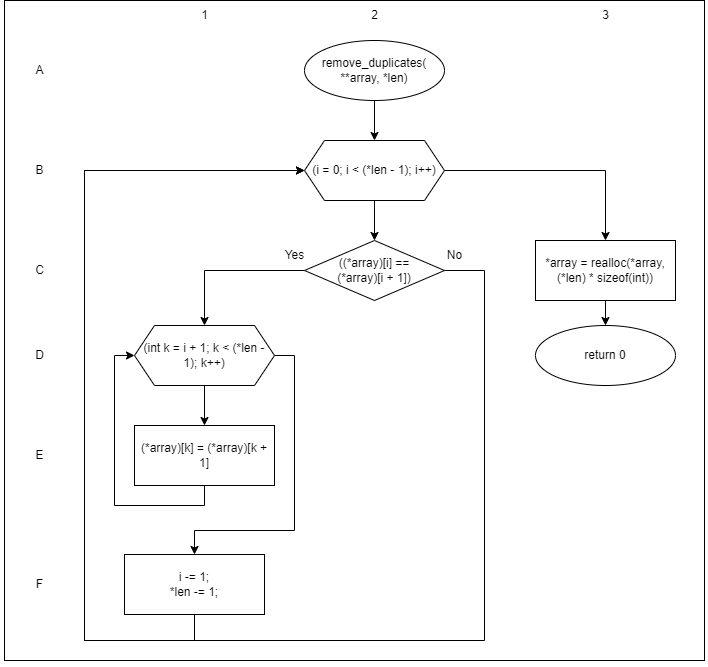


Рис. 1: Блок­схема алгоритма работы функции bubble\_sort()

Рис. 2: Блок-схема алгоритма работы функции remove\_duplicates()

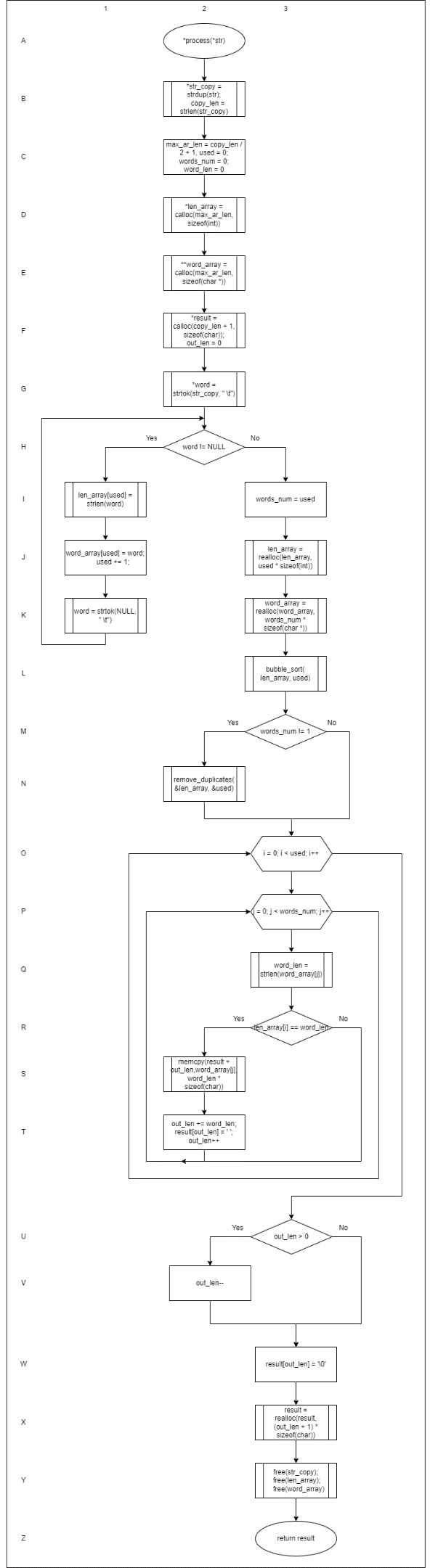
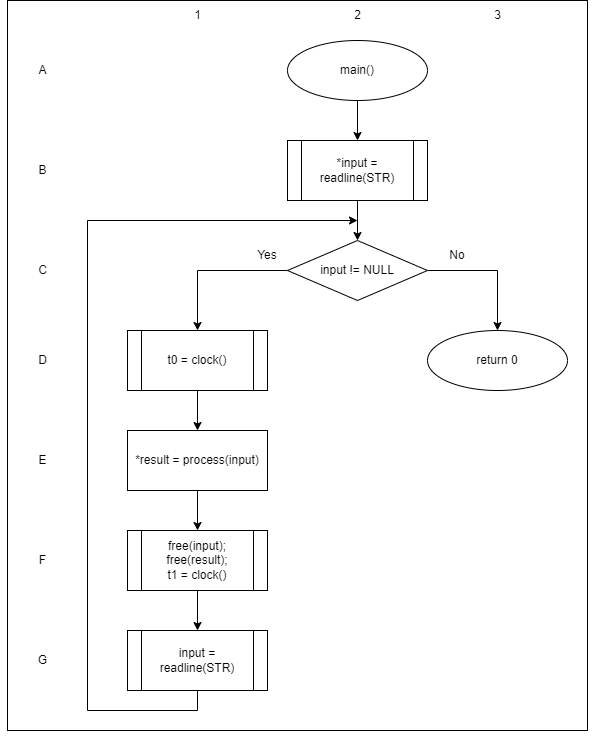
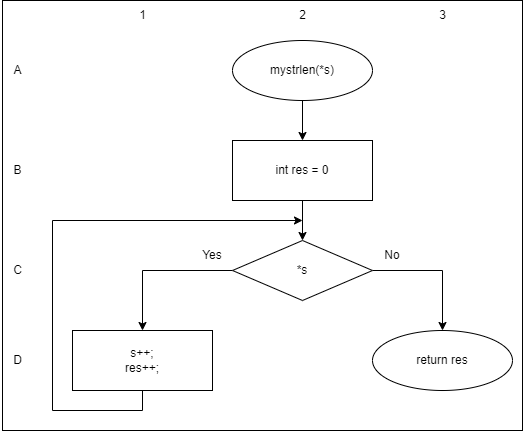
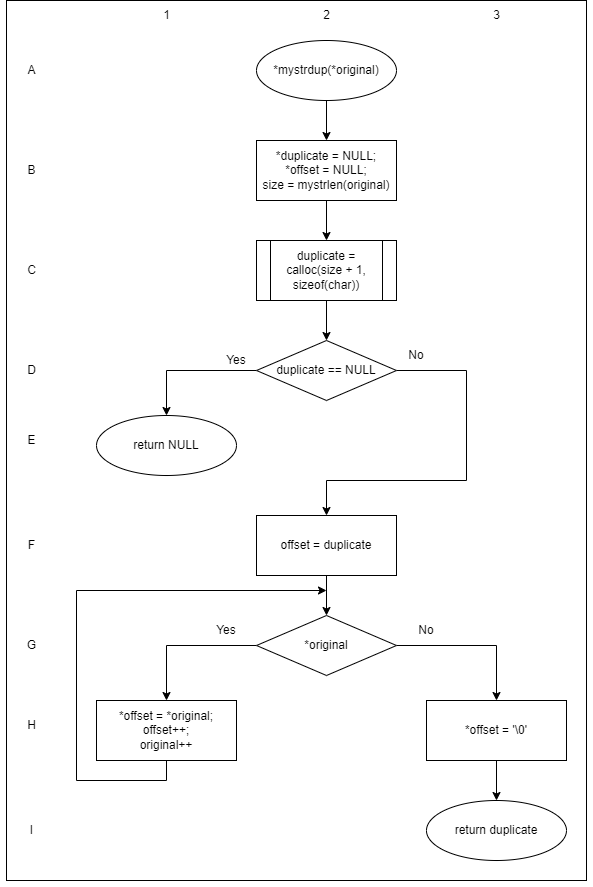


Рис. 3: Блок-схема алгоритма работы функции process()

Рис. 4: Блок-схема алгоритма работы функции main()



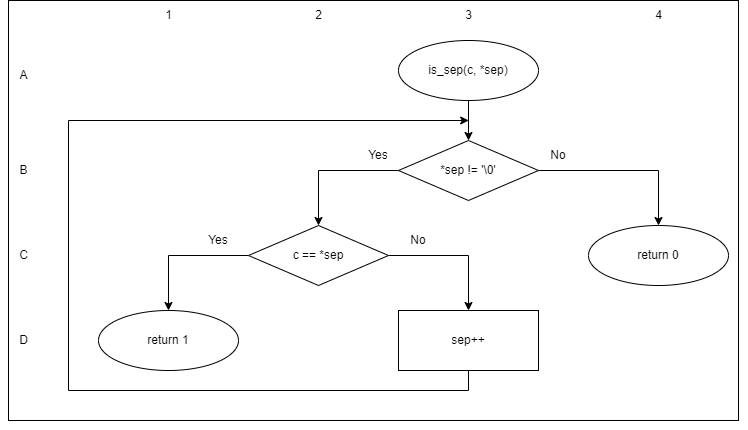


Рис. 5: Блок-схема алгоритма работы функции mystrlen()

Рис. 6: Блок-схема алгоритма работы функции mystrdup()

Рис. 7: Блок-схема алгоритма работы функции is\_sep()

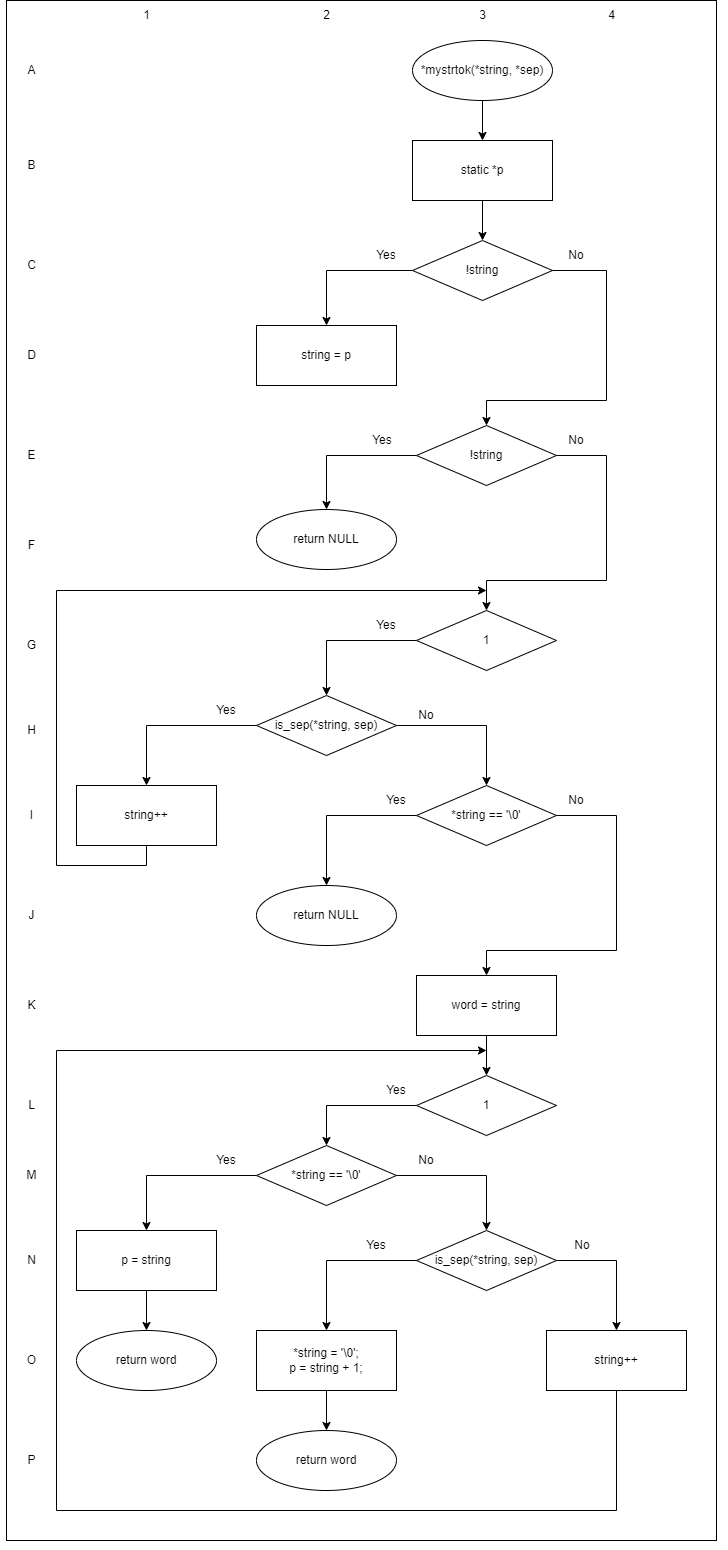
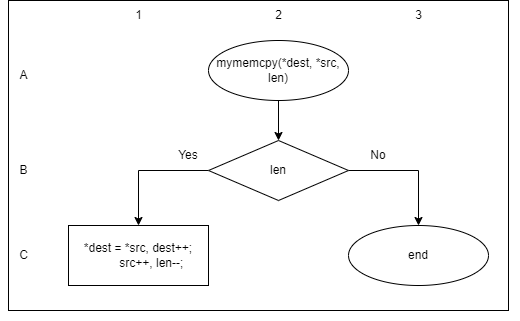


Рис. 8: Блок-схема алгоритма работы функции mystrtok()

Рис. 9: Блок-схема алгоритма работы функции mymemcpy()

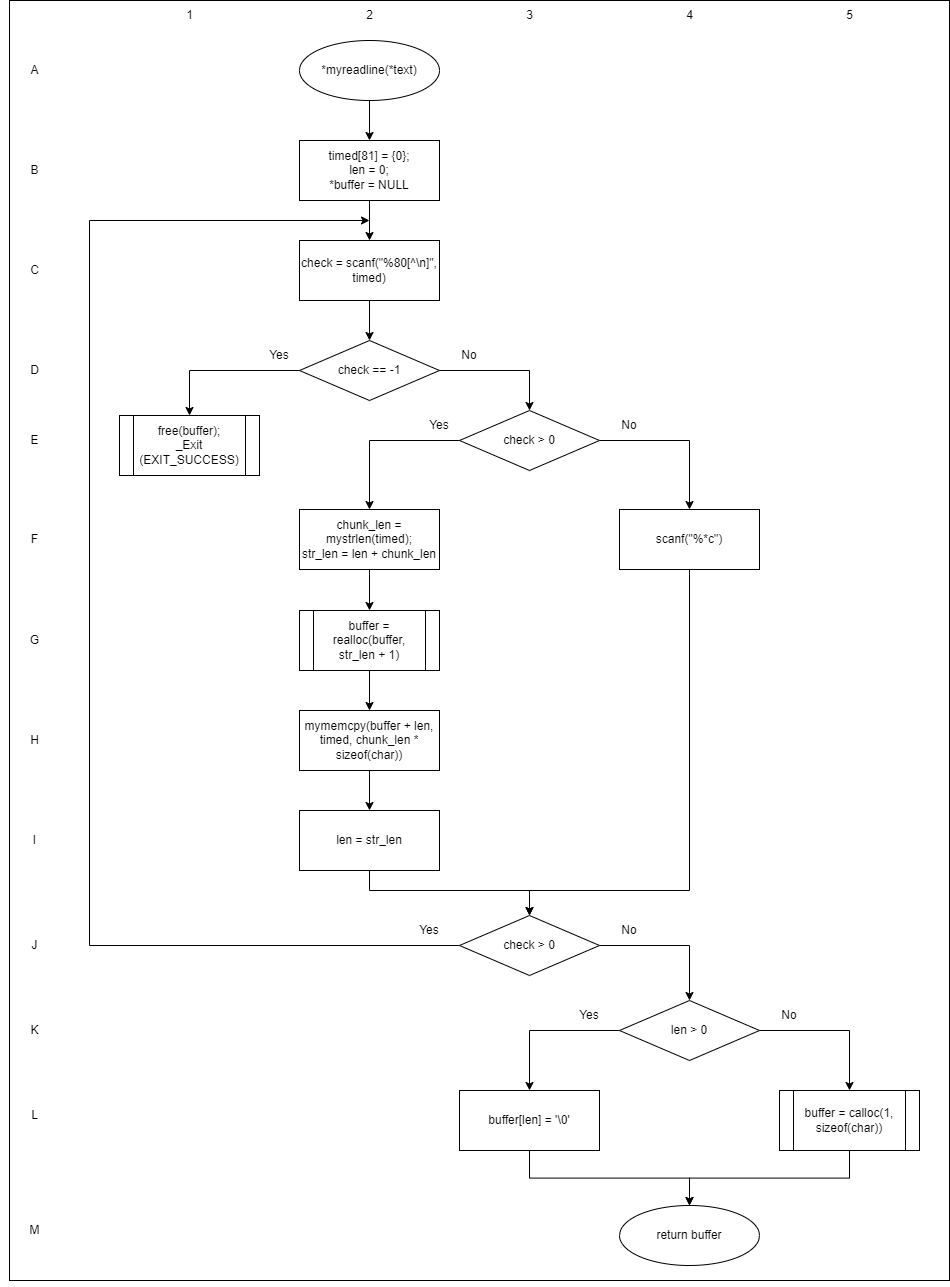
****

Рис. 10: Блок-схема алгоритма работы функции myreadline()

**4. Исходные коды разработанных программ**

Листинг 1: Исходные коды программы lab4 (1)

1) Файл: main.c

#include <stdio.h>  
#include <stdlib.h>  
#include <readline/readline.h>  
#include <string.h>  
#include <time.h>  
  
#define STR "Your input: "  
  
int bubble\_sort(int \*array, int len) {  
 for (int i = 0; i < len; i++) {  
 for (int j = len - 1; j > i; j--) {  
 if (array[j] < array[j - 1]) {  
 int t = array[j - 1];  
 array[j - 1] = array[j];  
 array[j] = t;  
 }  
 }  
 }  
 return 0;  
}  
  
int remove\_duplicates(int \*\*array, int \*len){  
 for (int i = 0; i < (\*len - 1); i++){  
 if ((\*array)[i] == (\*array)[i + 1]){  
 for (int k = i + 1; k < (\*len - 1); k++){  
 (\*array)[k] = (\*array)[k + 1];  
 }  
 i -= 1;  
 \*len -= 1;  
 }  
 }  
  
 \*array = realloc(\*array, (\*len) \* sizeof(int));  
  
 return 0;  
}  
  
char \*process(const char \*str) {  
 char \*str\_copy = strdup(str);  
 int copy\_len = strlen(str\_copy);  
 int max\_ar\_len = copy\_len / 2 + 1;  
 int used = 0;  
 int words\_num = 0;  
 int word\_len = 0;  
 int \*len\_array = calloc(max\_ar\_len, sizeof(int));  
 char \*\*word\_array = calloc(max\_ar\_len, sizeof(char \*));  
 char \*result = calloc(copy\_len + 1, sizeof(char));  
 int out\_len = 0;  
 char \*word = strtok(str\_copy, " \t");  
 while (word != NULL) {  
 len\_array[used] = strlen(word);  
 word\_array[used] = word;  
 used += 1;  
 word = strtok(NULL, " \t");  
 }  
 words\_num = used;  
 len\_array = realloc(len\_array, used \* sizeof(int));  
 word\_array = realloc(word\_array, words\_num \* sizeof(char \*));  
 bubble\_sort(len\_array, used);  
 if (words\_num != 1){  
 remove\_duplicates(&len\_array, &used);  
 }  
  
 for (int i = 0; i < used; i++){  
 for (int j = 0; j < words\_num; j++){  
 word\_len = strlen(word\_array[j]);  
 if (len\_array[i] == word\_len){  
 memcpy(result + out\_len, word\_array[j], word\_len \* sizeof(char));  
 out\_len += word\_len;  
 result[out\_len] = ' ';  
 out\_len++;  
 }  
 }  
 }  
  
 if (out\_len > 0) {  
 out\_len--;  
 }  
  
 result[out\_len] = '\0';  
 result = realloc(result, (out\_len + 1) \* sizeof(char));  
 free(str\_copy);  
 free(len\_array);  
 free(word\_array);  
  
 return result;  
}  
  
int main() {  
 char \*input = readline(STR);  
 while (input != NULL) {  
 clock\_t t0 = clock();  
 printf("\"%s\"\n", input);  
 char \*result = process(input);  
 printf("\"%s\"\n", result);  
 free(input);  
 free(result);  
 clock\_t t1 = clock();  
 printf("%lf\n", (double)(t1 - t0) / CLOCKS\_PER\_SEC);  
 input = readline(STR);  
 }  
 return 0;  
}

Листинг 2: Исходные коды программы lab4 (2)

1) Файл: main.c

#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#include "read.h"  
  
#define STR "Your input: "  
  
int bubble\_sort(int \*array, int len){  
 for (int i = 0; i < len; i++) {  
 for (int j = len - 1; j > i; j--) {  
 if (array[j] < array[j - 1]) {  
 int t = array[j - 1];  
 array[j - 1] = array[j];  
 array[j] = t;  
 }  
 }  
 }  
 return 0;  
}  
  
int remove\_duplicates(int \*\*array, int \*len){  
 for (int i = 0; i < (\*len - 1); i++){  
 if ((\*array)[i] == (\*array)[i + 1]){  
 for (int k = i + 1; k < (\*len - 1); k++){  
 (\*array)[k] = (\*array)[k + 1];  
 }  
 i -= 1;  
 \*len -= 1;  
 }  
 }  
  
 \*array = realloc(\*array, (\*len) \* sizeof(int));  
  
 return 0;  
}  
  
char \*process(const char \*str){  
 char \*str\_copy = mystrdup(str);  
 int copy\_len = mystrlen(str\_copy);  
 int max\_ar\_len = copy\_len / 2 + 1;  
 int used = 0;  
 int words\_num = 0;  
 int word\_len = 0;  
 int \*len\_array = calloc(max\_ar\_len, sizeof(int));  
 char \*\*word\_array = calloc(max\_ar\_len, sizeof(char \*));  
 char \*result = calloc(copy\_len + 1, sizeof(char));  
 int out\_len = 0;  
 char \*word = mystrtok(str\_copy, " \t");  
 while (word != NULL){  
 len\_array[used] = mystrlen(word);  
 word\_array[used] = word;  
 used += 1;  
 word = mystrtok(NULL, " \t");  
 }  
 words\_num = used;  
 len\_array = realloc(len\_array, used \* sizeof(int));  
 word\_array = realloc(word\_array, words\_num \* sizeof(char \*));  
 bubble\_sort(len\_array, used);  
 if (words\_num != 1){  
 remove\_duplicates(&len\_array, &used);  
 }  
  
 for (int i = 0; i < used; i++){  
 for (int j = 0; j < words\_num; j++){  
 word\_len = mystrlen(word\_array[j]);  
 if (len\_array[i] == word\_len){  
 mymemcpy(result + out\_len, word\_array[j], word\_len \* sizeof(char));  
 out\_len += word\_len;  
 result[out\_len] = ' ';  
 out\_len++;  
 }  
 }  
 }  
  
 if (out\_len > 0){  
 out\_len--;  
 }  
  
 result[out\_len] = '\0';  
 result = realloc(result, (out\_len + 1) \* sizeof(char));  
 free(str\_copy);  
 free(len\_array);  
 free(word\_array);  
  
 return result;  
}  
  
int main(){  
 char \*input = myreadline(STR);  
 while (input != NULL){  
 clock\_t t0 = clock();  
 printf("\"%s\"\n", input);  
 char \*result = process(input);  
 printf("\"%s\"\n", result);  
 free(input);  
 free(result);  
 clock\_t t1 = clock();  
 printf("%lf\n", (double)(t1 - t0) / CLOCKS\_PER\_SEC);  
 input = myreadline(STR);  
 }  
 return 0;  
}

2) Файл: read.c

#include <stdio.h>  
#include <stdlib.h>  
  
size\_t mystrlen(const char \*s) {  
 int res = 0;  
 while (\*s) {  
 s++;  
 res++;  
 }  
 return res;  
}  
  
char \*mystrdup(const char \*original){  
 char \*duplicate = NULL;  
 char \*offset = NULL;  
 int size = mystrlen(original);  
   
 duplicate = calloc(size + 1, sizeof(char));  
   
 if (duplicate == NULL){  
 return NULL;  
 }  
   
 offset = duplicate; // чтоб сохранить исходный указатель в duplicate  
   
 while (\*original){  
 \*offset = \*original;  
 offset++;  
 original++;  
 }  
 \*offset = '\0';  
  
 return duplicate;  
}  
  
int is\_sep(char c, const char \*sep){  
 while(\*sep != '\0'){  
 if(c == \*sep){  
 return 1;  
 }  
 sep++;  
 }  
 return 0;  
}  
  
char \*mystrtok(char \*string, const char \*sep){  
 static char \*p;  
 char \*word;  
  
 if (!string){  
 string = p;  
 }  
  
 if (!string){  
 return NULL;  
 }  
  
 while (1){  
 if (is\_sep(\*string, sep)){ // Если нет разделителей в начале, то break  
 string++;  
 continue;  
 }  
 if (\*string == '\0'){  
 return NULL;  
 }   
 break;  
 }  
  
 word = string;  
   
 while (1){  
 if (\*string == '\0'){ // Конец строки  
 p = string;  
 return word;  
 }  
 if (is\_sep(\*string, sep)){  
 \*string = '\0';  
 p = string + 1;  
 return word;  
 }  
 string++;  
 }  
}  
  
void mymemcpy (char \*dest, const char \*src, size\_t len){  
 while (len){  
 \*dest = \*src;  
 dest++;  
 src++;  
 len--;  
 }  
}  
  
char \*myreadline(const char \*text){  
 printf("%s", text);  
 char timed[81] = {0};  
 int len = 0;  
 int check;  
 char \*buffer = NULL;  
  
 do {  
 check = scanf("%80[^\n]", timed);  
  
 if (check == -1){  
 free(buffer);  
 printf("Program has been stopped\n");  
 \_Exit (EXIT\_SUCCESS);  
 }  
 else if (check > 0){  
 int chunk\_len = mystrlen(timed);  
 int str\_len = len + chunk\_len;  
 buffer = realloc(buffer, str\_len + 1);  
 mymemcpy(buffer + len, timed, chunk\_len \* sizeof(char));  
 len = str\_len;  
 }  
 else {  
 scanf("%\*c");  
 }  
 }  
 while (check > 0);  
 if (len > 0){  
 buffer[len] = '\0';  
 }  
 else {  
 buffer = calloc(1, sizeof(char));  
 }  
 return buffer;  
}

3) Файл: read.h

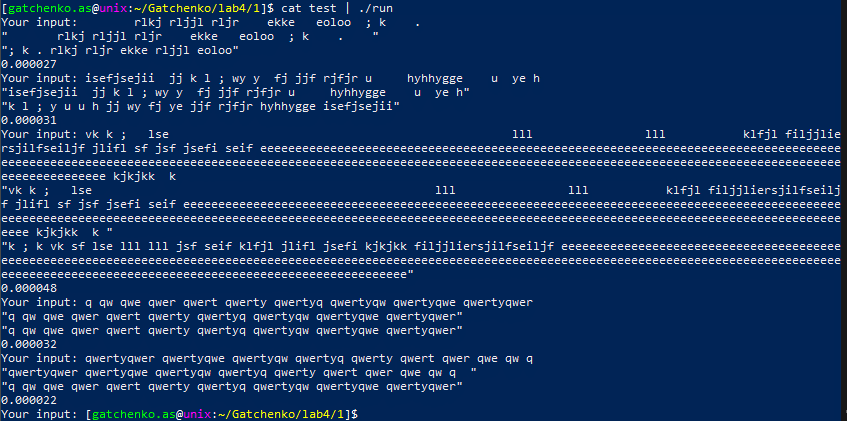
#ifndef READ  
#define READ  
#include <stddef.h>  
  
size\_t mystrlen(const char \*s);  
char \*mystrdup(const char \*original);  
char \*mystrtok(char \*string, const char \*sep);  
char \*mymemcpy (char \*dest, const char \*src, size\_t len);  
char \*myreadline(const char \*text);  
  
#endif

**5. Описание тестовых примеров**

Таблица 1: Тестовые примеры lab4 (1) и lab4 (2)

|  |  |  |  |
| --- | --- | --- | --- |
| Введенная строка | Строка после обработки | Время выполнения в lab4 (1) (сек) | Время выполнения в lab4 (2) (сек) |
| « rlkj rljjl rljr ekke eoloo ; k . » | «; k . rlkj rljr ekke rljjl eoloo» | 0.000027 | 0.000039 |
| «vk k ; lse lll lll klfjl filjjliersjilfseiljf jlifl sf jsf jsefi seif eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee kjkjkk k» | «k ; k vk sf lse lll lll jsf seif klfjl jlifl jsefi kjkjkk filjjliersjilfseiljf eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee» | 0.000048 | 0.000075 |
| «q qw qwe qwer qwert qwerty qwertyq qwertyqw qwertyqwe qwertyqwer» | «q qw qwe qwer qwert qwerty qwertyq qwertyqw qwertyqwe qwertyqwer» | 0.000032 | 0.000049 |
| «qwertyqwer qwertyqwe qwertyqw qwertyq qwerty qwert qwer qwe qw q » | «q qw qwe qwer qwert qwerty qwertyq qwertyqw qwertyqwe qwertyqwer» | 0.000022 | 0.000024 |

**6. Скриншоты**



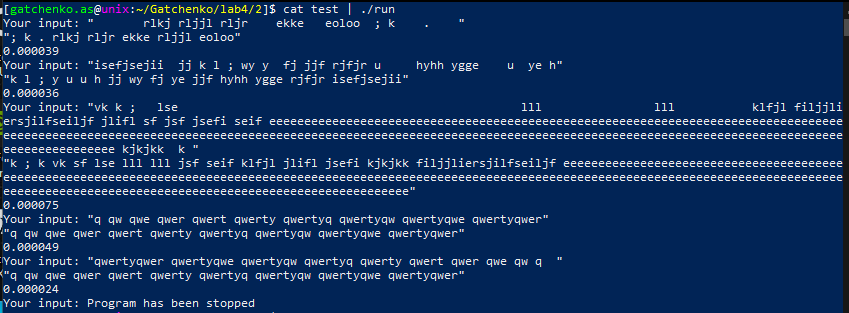


Рис. 11: Запуск программы lab4 (1)

Рис. 12: Запуск программы lab4 (2)

**7. Выводы**

В ходе выполнения данной работы на примере программы, выполняющей обработку строк, были рассмотрены базовые принципы работы построения программ на языке C и обработки строк:

1. Организация ввода/вывода, а также проверка корректности ввода.
2. Разработка функций.
3. Объявление и использование переменных.
4. Выполнение простейших арифметических операций над целочисленными и дробными операндами.
5. Использование циклов и условий.
6. Использование указателей (параметров) на символы, а также на массивы символов.
7. Разбиение программы на несколько файлов.
8. Осуществление ввода с помощью сторонней библиотеки GNU readline.
9. Работа с памятью.