

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Сибирский государственный университет телекоммуникаций и  
информатики»  
(СибГУТИ)

Факультет ИВТ  
Кафедра вычислительных систем

**Курсовая работа**  
на тему «Обработка последовательной информации»  
Вариант 1.5  
«Разработка простейшего переводчик»

Выполнил:  
студент гр. ИВ-122  
Клепче Г.В.

Проверил:  
Ст. преп. Кафедры ВС  
Фульман В.О.

Новосибирск 2022

## СОДЕРЖАНИЕ

Тема курсовой работы.....	3
Задание на курсовую работу.....	3
Анализ задачи.....	4
Тестовые данные.....	10
Листинг программы.....	11

## Тема курсовой работы

Реализовать простейший переводчик.

### Задание на курсовую работу

Разработать программу **translate**, выполняющую перевод текста с помощью словаря. Команда **translate** принимает на вход 3 файла. Первый содержит исходный текст, который необходимо перевести. Второй файл имеет вид простейшего словаря, где каждому слову на исходном языке соответствует слово на целевом. Третий файл необходимо создать и записать в него результат работы переводчика.

Формат исходного текста должен быть сохранен.

Критерии оценки:

Оценка «удовлетворительно»: не реализована поддержка файласловаря, словарь задается статически в программе. Не предусмотрено динамическое выделение памяти под входные данные.

Оценка «хорошо»: программа реализована в полном соответствии с заданием.

Обязательно динамическое выделение памяти под входные данные.

Оценка «отлично»: не предусмотрена, может быть предложен свой вариант усложнения.

### Анализ задачи

Запуск программы должен производиться со следующими аргументами командной строки:

```
$ translate text_rus.txt dictionary.txt text_eng.txt
```

Программа должна перевести текст в файле *text\_rus.txt* с помощью словаря *dictionary.txt*, и записать результат в файл *text\_eng.txt*. Примерное содержимое файлов и результат работы программы представлен на рисунке.

*Text\_rus.txt:*

Тигр,  
    Тигр,  
жгучий страх.  
Ты горюшь в ночных лесах.  
    Чей бессмертный взор,  
любя,  
Создал страшного тебя?

*Dictionary.txt:*

Тигр - Tyger  
Страх - Fear  
Ты - You  
Взор - Eye

*Text\_eng.txt:*

Tyger,  
    Tyger,  
жгучий fear.  
You горюшь в ночных лесах.  
    Чей бессмертный Eye,  
любя,  
Создал страшного тебя?

Форматирование текста в файле *text\_rus.txt* должно быть сохранено и в итоговом файле *text\_eng.txt*, т.е. сохранены все сдвиги и переносы по тексту. Задание не предусматривает поиск однокоренных слов, поэтому замена слова происходит только по полному соответствию

## Псевдокод

```
DECLARE * source
DECLARE Nsource
DECLARE * result
DECLARE NResult
DECLARE ** dict_1
DECLARE ** dict_2
DECLARE NDict
```

Function Declaration ->void LoadDictionary(char \* filename) //Загрузить словарь

Function Declaration ->void LoadSource(char \* filename)//Загрузить исходный текст

Function Declaration ->void Translate()//Сформировать результат

Function Declaration ->void SaveResult(char \* filename)//Сохранить результат

```
ALGORITHM main(narg<integer>,args<unknown type(char**)>)
```

```
Begin
```

```
if (narg != 4)
```

```
    Begin
```

```
        PRINT "usage : translate text_rus dictionary text_eng"
```

```
    getchar()
```

```
    return 1
```

```
End
```

```
LoadDictionary(args(2)) //Загрузить словарь
```

```
LoadSource(args(1))//Загрузить исходный текст
```

```
result = malloc(2 * Nsource) //С двойным запасом
```

```
Translate()//Сформировать результат
```

```
SaveResult(args(3))//Сохранить результат
```

```
free(result) //Освободить память
```

```
free(dict_1)
```

```

free(dict_2)

End
ALGORITHM LoadSource(*<character>)
Begin
FILE * F = fopen(filename, "rb")
if (!F)
    Begin
Nsource = 0
    PRINT "No file ",filename
return

End
fseek(F, 0, SEEK_END)
Nsource = ftell(F)
source = malloc(Nsource)
fseek(F, 0, SEEK_SET)
fread(source, 1, Nsource, F)
fclose(F)
    PRINT "Loaded ",filename

End

ALGORITHM SaveResult(*<character>)
Begin
FILE * F = fopen(filename, "wb")
fwrite(result, 1, NResult, F)
fclose(F)
    PRINT "Saved ", filename

End
ALGORITHM PrintDict()
Begin
    DECLARE k AS INTEGER
    FOR k = 1 TO NDict STEP 1
        PRINT " dict_1(k) —,"- ", dict_2(k)"

End
ALGORITHM LoadDictionary(*<character>)

```

```

Begin
FILE * F = fopen(filename, "r")
NDict = 0
if (!F)
    Begin
        PRINT "No file ", filename
    return
End

    DECLARE s(100) AS CHARACTER
while (1)
    Begin
fgets(s,sizeof(s),F)
if (feof(F)) break
        NDict = NDict+ 1
    End
fclose(F)
dict_1 = malloc(NDict*sizeof(char*))
dict_2 = malloc(NDict*sizeof(char*))
F = fopen(filename, "r")
    DECLARE k AS INTEGER
    FOR k = 1 TO NDict STEP 1
fgets(s, sizeof(s), F)
        DECLARE w1from = 0, w1to=0, w2from = 0, w2to = 0 AS INTEGER
while (s(w1from) = ' ')
    Begin
        w1from = w1from+ 1
    End
w1to = w1from
while (s(w1to) != ' ')
    Begin
        w1to = w1to+ 1
    End
w2from = w1to
while (s(w2from) = ' ')
    Begin
        w2from = w2from+ 1
    End

```

```

w2to = w2from
while (s(w2to) != ' ' && s(w2to) != '\n' && s(w2to) != '\r' && s(w2to) != '\0')
    Begin
        w2to = w2to + 1
    End
dict_1(k) = malloc(w1to - w1from + 1) memset(dict_1(k), '\0', w1to - w1from + 1)
memcpy(dict_1(k), &s(w1from), w1to - w1from)
dict_2(k) = malloc(w2to - w2from + 1) memset(dict_2(k), '\0', w2to - w2from + 1)
memcpy(dict_2(k), &s(w2from), w2to - w2from)

End
fclose(F)

End

const char alphabetEng() = "abcdefghijklmnopqrstuvwxyz"
const char AlphabetEng() = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
const char alphabetCyr() = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя"
const char AlphabetCyr() = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"

End
pos = strchr(AlphabetEng, c)
if (pos)
    Begin
        return *(alphabetEng + (pos - AlphabetEng))
    End
return c

End

ALGORITHM to_upper(c<character>)
Begin
    DECLARE * pos = strchr(alphabetCyr, c) AS CHARACTER
    if (pos)
        Begin
            return *(AlphabetCyr + (pos - alphabetCyr))
        End
End

```

```

pos = strchr(alphabetEng, c)
if (pos)
    Begin
return *(AlphabetEng + (pos - alphabetEng))

```

```

End
return c
End

```

```

ALGORITHM is_upper(c<character>)
Begin
if (strchr(AlphabetCyr,c))
return 1
if (strchr(AlphabetEng,c))
return 1
return 0

```

```

End

```

```

ALGORITHM Compare(*<character>,*<character>)
Begin
if (strlen(word1) != strlen(word2)) return -1
    DECLARE k AS INTEGER

```

```

ALGORITHM set_First_Letter(*<character>,*<character>)
Begin
if (is_upper(source(0)))
result(0) = to_upper(result(0))
    elseresult(0) = to_lower(result(0))

```

```

End

```

```

ALGORITHM TranslateWord(*<character>,*<character>)
Begin
    DECLARE found = -1 AS INTEGER
    DECLARE k AS INTEGER
    FOR k = 1 TO NDict STEP 1 if (Compare(source, dict_1(k))=0)
        Begin
found = k

```



```

break

End

if (found >= 0)
    Begin
    strcpy(result, dict_2(found))
    set_First_Letter(source, result)
    else strcpy(result, source)

End

ALGORITHM TranslateWord(*<character>,*<character>)
Begin
    DECLARE found = -1 AS INTEGER
    DECLARE k AS INTEGER
    FOR k = 1 TO NDict STEP 1 if (Compare(source, dict_1(k))=0)
        Begin
        found = k
        break

End

if (found >= 0)
    Begin
    strcpy(result, dict_2(found))
    set_First_Letter(source, result)
    else strcpy(result, source)

End

ALGORITHM Translate()
Begin

    DECLARE p = 0
    DECLARE word(100) = ""
    DECLARE trans(100) = ""
    DECLARE k AS INTEGER
    FOR k = 1 TO Nsource STEP 1

```

```

if (!is_Letter(source(k)))
    Begin
if (strlen(word))
    Begin
TranslateWord(word, trans) //перевести

```

## Тестовые данные

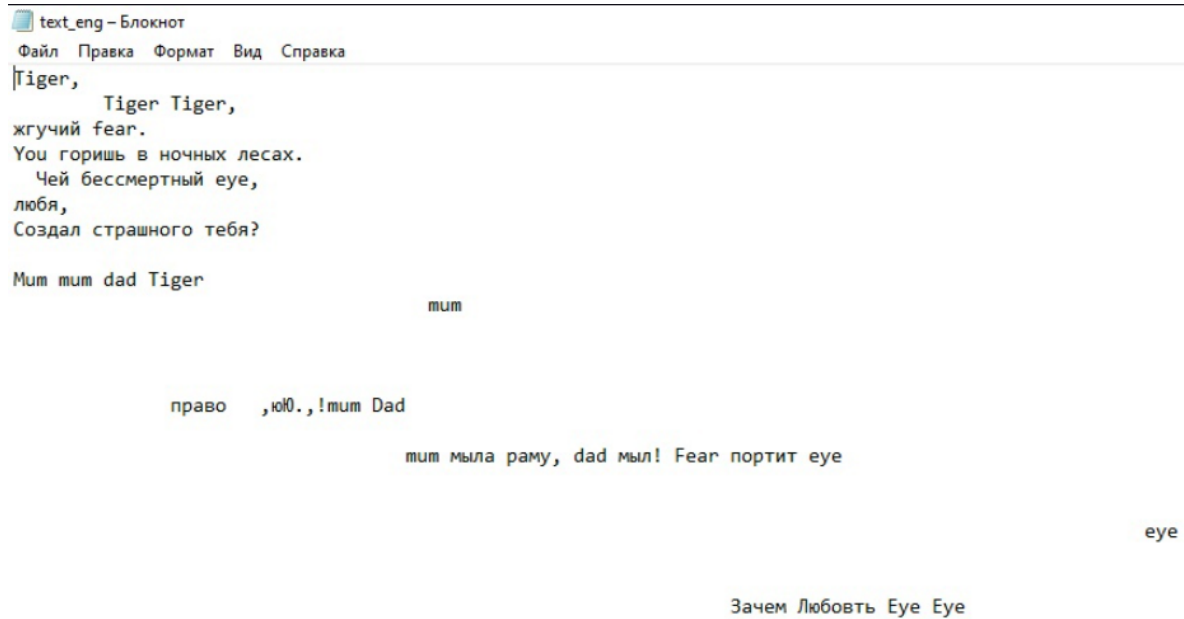
Попробуем перевести текстовый файл text\_rus.txt:



Со словарем dictionary.txt



Получаем перевод text\_eng.txt:



text\_eng - Блокнот  
Файл Правка Формат Вид Справка

Tiger,  
Tiger Tiger,  
жгучий fear.  
You горишь в ночных лесах.  
Чей бессмертный eye,  
любя,  
Создал страшного тебя?

Mum mum dad Tiger

mum

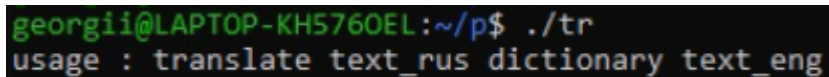
право ,юю.,!mum Dad

mum мыла раму, dad мыл! Fear портит eye

eye

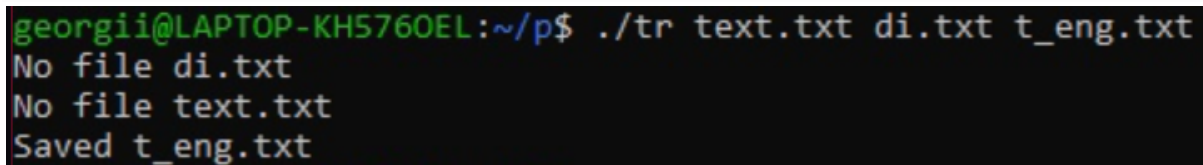
Зачем Любовь Eye Eye

Запустим программу без аргументов.



```
georgii@LAPTOP-KH5760EL:~/p$ ./tr
usage : translate text_rus dictionary text_eng
```

Запустим программу с не существующими файлами:



```
georgii@LAPTOP-KH5760EL:~/p$ ./tr text.txt di.txt t_eng.txt
No file di.txt
No file text.txt
Saved t_eng.txt
```

## ПРИЛОЖЕНИЕ

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  char * source; //Исходный текст
6  int Nsource; //Длина исходного текста
7  char * result; //Результат подстрочного перевода
8  int NResult; //Длина результирующего текста
9  char ** dict_1; //Словарь - исходный язык
10 char ** dict_2; //Словарь - результирующий язык
11 int NDict; //Количество слов в словаре
12
13 void LoadDictionary(char * filename); //Загрузить словарь
14 void LoadSource(char * filename); //Загрузить исходный текст
15 void Translate(); //Сформировать результат
16 void SaveResult(char * filename); //Сохранить результат
17
18 int main(int nargs, char** args)
19 {
20     if (nargs != 4) {
21         printf("usage : translate text_rus dictionary text_eng\n");
22         getchar();
23         return 1;
24     }
25
26     LoadDictionary(args[2]);
27     LoadSource(args[1]);
28     result = malloc(2 * Nsource);
29     Translate();
30     SaveResult(args[3]);
31     free(result);
32     free(dict_1);
33     free(dict_2);
34 }
35
36 void LoadSource(char * filename)
37 {
38     FILE * F = fopen(filename, "rb");
39     if (!F) {
40         Nsource = 0;
41         printf("No file %s\n", filename);
42         return;
43     }
44     fseek(F, 0, SEEK_END);
45     Nsource = ftell(F);
46     source = malloc(Nsource);
47     fseek(F, 0, SEEK_SET);
48     fread(source, 1, Nsource, F);
49     fclose(F);
50     printf("Loaded %s\n", filename);
51 }
52
53 void SaveResult(char * filename)
54 {

```

```

55 FILE * F = fopen(filename, "wb");
56 fwrite(result, 1, NResult, F);
57 fclose(F);
58 printf("Saved %s\n", filename);
59 }
60
61 void PrintDict()
62 {
63     system("chcp 1251");
64     int k;
65     for (k = 0; k < NDict; k++)
66         printf("%s -- %s\n", dict_1[k], dict_2[k]);
67 }
68
69 void LoadDictionary(char * filename)
70 {
71     FILE * F = fopen(filename, "r");
72     NDict = 0;
73     if (!F) {
74         printf("No file %s\n", filename);
75         return;
76     }
77
78     char s[100];
79     while (1) {
80         fgets(s, sizeof(s), F);
81         if (feof(F)) break;
82         NDict++;
83     }
84     fclose(F);
85     dict_1 = malloc(NDict * sizeof(char*));
86     dict_2 = malloc(NDict * sizeof(char*));
87     F = fopen(filename, "r");
88
89     int k;
90     for (k = 0; k < NDict; k++) {
91         fgets(s, sizeof(s), F);
92         int w1from = 0, w1to = 0, w2from = 0, w2to = 0;
93         while (s[w1from] == ' ') {
94             w1from++;
95         }
96         w1to = w1from;
97         while (s[w1to] != ' ') {
98             w1to++;
99         }
100         w2from = w1to;
101         while (s[w2from] == ' ') {
102             w2from++;
103         }
104         w2to = w2from;
105         while (s[w2to] != ' ' && s[w2to] != '\n' && s[w2to] != '\r' && s[w2to] != '\0') {
106             w2to++;
107         }
108         dict_1[k] = malloc(w1to - w1from + 1); memset(dict_1[k], '\0', w1to - w1from + 1);

```

```

109 memcpy(dict_1[k], &s[w1from], w1to - w1from);
110 dict_2[k] = malloc(w2to - w2from + 1); memset(dict_2[k], '\0', w2to - w2from + 1);
111 memcpy(dict_2[k], &s[w2from], w2to - w2from);
112 }
113 fclose(F);
114 //PrintDict(); getchar();
115 }
116
117 const char alphabetEng[] = "abcdefghijklmnopqrstuvwxyz";
118 const char AlphabetEng[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
119 const char alphabetCyr[] = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя";
120 const char AlphabetCyr[] = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";
121
122 int is_Letter(char c)
123 {
124     if (strchr(alphabetEng, c))
125         return 1; //Если содержит - то это буква
126     if (strchr(AlphabetEng, c))
127         return 1;
128     if (strchr(alphabetCyr, c))
129         return 1;
130     if (strchr(AlphabetCyr, c))
131         return 1;
132     return 0; //Не буква
133 }
134
135 char to_lower(char c)
136 {
137     char * pos = strchr(AlphabetCyr, c);
138     if (pos) {
139         return *(alphabetCyr + (pos - AlphabetCyr));
140     }
141     pos = strchr(AlphabetEng, c);
142     if (pos) {
143         return *(alphabetEng + (pos - AlphabetEng));
144     }
145     return c;
146 }
147
148 char to_upper(char c)
149 {
150     char * pos = strchr(alphabetCyr, c);
151     if (pos) {
152         return *(AlphabetCyr + (pos - alphabetCyr));
153     }
154     pos = strchr(alphabetEng, c);
155     if (pos) {
156         return *(AlphabetEng + (pos - alphabetEng));
157     }
158     return c;
159 }
160
161
162 int is_upper(char c)

```

```

163 {
164     if (strchr(AlphabetCyr,c))
165         return 1;
166     if (strchr(AlphabetEng,c))
167         return 1;
168     return 0;
169 }
170
171 //Сравнить два слова без учета регистра
172 int
173 Compare(char * word1, char * word2)
174 {
175     if (strlen(word1) != strlen(word2)) return -1; //не равны
176     int k;
177     for (k = 0; k < strlen(word1); k++)
178         if (to_lower(word1[k]) != to_lower(word2[k])) return -1;
179     return 0; //Равны
180 }
181
182 //Установить первую букву
183 void set_First_Letter(char * source, char * result)
184 {
185     if (is_upper(source[0]))
186         result[0] = to_upper(result[0]);
187     else
188         result[0] = to_lower(result[0]);
189 }
190
191 void TranslateWord(char * source, char * result)
192 {
193     //Найти в словаре
194     int found = -1;
195     int k;
196     for (k = 0; k < NDict; k++)
197         if (Compare(source, dict_1[k])==0) {
198             found = k;
199             break;
200         }
201
202     if (found >= 0) {
203         strcpy(result, dict_2[found]);
204         set_First_Letter(source, result);
205     } else
206         strcpy(result, source);
207 }
208
209 void Translate()//Сформировать результат
210 {
211     int p = 0; //Позиция в переводе
212     char word[100] = ""; //Место для накопления слова
213     char trans[100] = ""; //Место для перевода
214     int k;
215     for (k = 0; k < Nsource; k++) {
216         //Если это пробелы и прочие знаки препинания

```

```

217  if (!is_Letter(source[k])) {
218      //И если есть "накопленное слово"
219      if (strlen(word)) {
220          //То перевести слово, поместить в результат, очистить накопленное
221          TranslateWord(word, trans); //перевести
222          //поместить в результат
223          int i;
224          for (i = 0; i < strlen(trans); i++)
225              result[p++] = trans[i];
226          word[0] = 0; //очистить накопленное
227      }
228      //и не забыть добавить знак препинания
229      result[p++] = source[k];
230  } else {
231      //Если же это буква, то добавить в накопленное слово
232      int L = strlen(word);
233      word[L] = source[k];
234      word[L + 1] = 0;
235  }
236  }
237  NResult = p;
238  }

```