

**Министерство образования и науки Российской Федерации**

**Федеральное агентство по образованию**

**Государственное образовательное учреждение высшего профессионального образования**

**«Московский государственный технический университет имени Н.Э. Баумана»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Робототехника и комплексная автоматизация» (РК)**

**Кафедра «Системы автоматизированного проектирования» (РК6)**

****

**Отчет по лабораторной работе №1 по курсу**

**«Операционные системы»**

**Студент: Сергеева Диана**

**Группа:** РК6-56Б

**Преподаватель:** Грошев С.В.

Проверил:

Дата:

2021 год

**13 вариант**

**Задание «Перекодировка текстовых файлов»:**

Нужно разработать программу перекодировки текстовых файлов OS UNIX, содержащих строчные и заглавные буквы алфавита кириллицы. Перекодировка должна осуществляться между кодовыми наборами КОИ-8 и СР1251. При этом исходная информация должна быть задана в текстовом файле, который составлен в кодировке КОИ-8. Результат перекодировки должен быть записан в другой текстовый файл, который создаётся для сохранения исходной информации в кодовом наборе СР1251. Имена исходного и результирующего файлов должны передаваться программе через аргументы командной строки её вызова. Чтение исходного и запись результирующего файлов должны осуществляться логическими блоками оптимального размера, при котором достигается максимальная эффективность операций ввода-вывод. Кроме того, в программе должен быть предусмотрен диагностический анализ возможности доступа к исходным и результирующим файлам в требуемых режимах, а также проверка совпадения их имён и обработка ситуаций, когда отсутствует заданный исходный файл и/или уже существует результирующий файл. При разработке программы необходимо использовать следующие системные вызовы OS UNIX: access, creat, open, read, write, close и exit.

Таблица кодировки должна формироваться в программе автоматически из таблиц символов, которые хранятся в текстовых файлах отдельно для исходной и результирующей кодировок. Имена таблиц, передаются программе как аргументы командной строки вместе с именами исходного и результирующего файлов.

**Исходный код:**

[#include](https://vk.com/im?sel=63386677&st=%23include) <unistd.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <errno.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <string.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <stdlib.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <fcntl.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <sys/stat.h>  
[#include](https://vk.com/im?sel=63386677&st=%23include) <sys/types.h>  
int diag(char \*, const char \*, int);  
unsigned char [int](https://vk.com/recode);  
static unsigned char buf[BUFSIZ]; //буфер-чтения - записи  
static unsigned char tab[32];  
void generate(int);  
int oct\_to\_dec(int);  
int main(int argc, char \*argv[])  
{  
int fds; //файловый дискриптор исходного КОИ-8 текста  
int fdt;  
int fdtWIN;  
int fdtKOI;  
int num;  
char unsigned tabKOI[32];  
char unsigned tabWIN[32];  
if (argc < 2)  
{  
exit(diag(argv[0], "Source & target file names?", 127));  
}  
if (argc < 3)  
{  
exit(diag(argv[0], "Target file names?", 63));  
}  
if (argc < 4)  
{  
exit(diag(argv[0], "Code table?", 65));  
}  
if ((fds = open(argv[1], 0)) < 0)  
{  
exit(diag(argv[1], strerror(errno), errno));  
}  
if ((fdtWIN = open(argv[3], 0)) < 0)  
{  
exit(diag(argv[1], strerror(errno), errno));  
}  
if ((fdtKOI = open(argv[4], 0)) < 0)  
{  
exit(diag(argv[1], strerror(errno), errno));  
}  
num=read(fdtKOI, buf, BUFSIZ);  
for (int i = 0; i < num; i++)  
{  
tabKOI[i] = buf[i]-192;  
}  
num=read(fdtWIN, buf, BUFSIZ);  
for (int i = 0; i < num; i++)  
{  
tabWIN[i] = buf[i]-224;  
}  
for (int i = 0; i < 32; i++)  
{  
tab[i] = tabWIN[i];  
}  
//------------------------------------  
if (access(argv[2], 0) == 0)  
exit(diag(argv[2], "Target file already exist", 255));  
errno = 0;  
if ((fdt = creat(argv[2], 0644)) < 0)  
exit(diag(argv[2], strerror(errno), errno));  
while ((num = read(fds, buf, BUFSIZ)) > 0)  
write(fdt, recode(num), num);  
close(fdt);  
close(fds);  
exit(0);  
}  
int diag(char \*name, const char \*mes, int code)  
{  
int len;  
strcpy(buf, name);  
strncat(buf, ": ", 2);  
strcat(buf, mes);  
len = strlen(buf);  
buf[len] = '\n';  
write(2, buf, len + 1);  
return (code);  
}  
unsigned char [int n](https://vk.com/recode)  
{  
unsigned char c;  
int i = 0;  
for (i = 0; i < n; i++)  
{  
c = buf[i];  
if (c < 192)  
{  
continue;  
}  
if (c < 224)  
{  
buf[i] = tab[c - 192] + 224;  
}  
else  
{  
buf[i] = tab[c - 224] + 192;  
}  
}  
return (buf);  
}  
void generate(int n)  
{  
int cur;  
int i = 0;  
int temp = 0;  
int tablePlace = 0;  
for (i = 0; i < n; i++)  
{  
if (buf[i] == ' ')  
{  
tab[tablePlace] = temp;  
temp = 0;  
tablePlace++;  
continue;  
}  
cur = buf[i] - '0';  
temp = temp \* 10 + cur;  
}  
}

**Результат работы программы:**



