НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

МОСКОВСКИЙ ЭНЕРГЕТИЧЕСКИЙ ИНСТИТУТ

Отчет к лабораторной работе №7

Дисциплины «Архитектура вычислительных систем»

Выполнил студент группы А-13-22

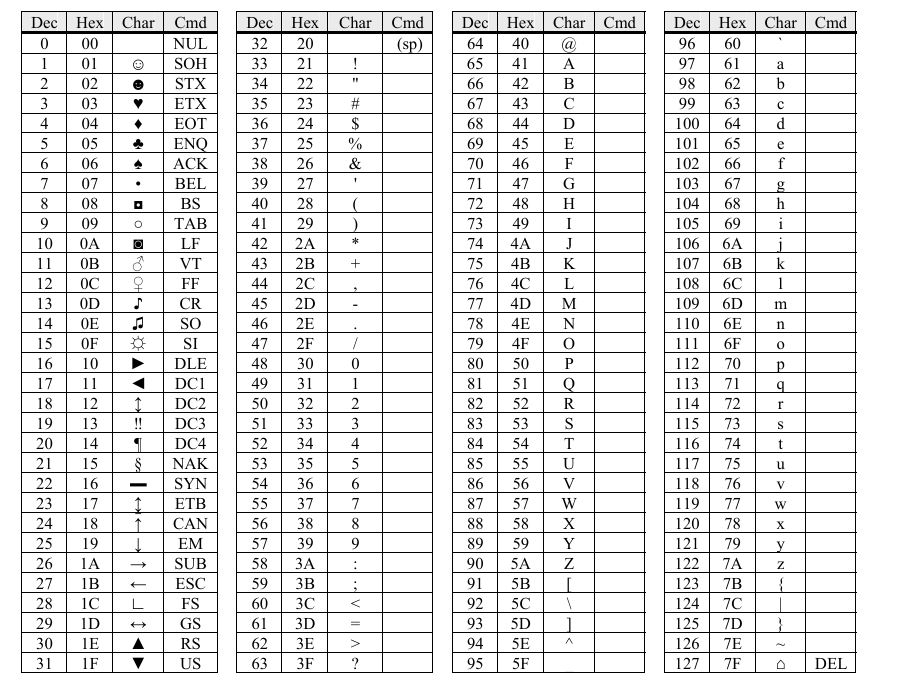
Кокляева Мария

Преподаватель: Шамаева О. Ю.

Москва 2024

## Лабораторное задание

* 1. **Запишите в двоичном и шестнадцатеричном коде английскую букву А.**



В **таблице** приведены **ASCII**-символы (Char) и их коды в десятичной (Dec) и шестнадцатиричной (Hex) системах счисления.

Из таблицы видно, что английская буква «А» =

* 1. **Запишите в двоичном и шестнадцатеричном коде в альтернативной кодировке русскую букву а.**

CP866 («Альтернативная кодировка») — набор символов, широко распространившийся во время господствования MS-DOS операционных систем. В ней записываются имена файлов в системе FAT, также кодировка CP866 (в отличие от [Windows-1251](https://kvodo.ru/tablitsa-windows-1251.html)) сохранила псевдографические символы.

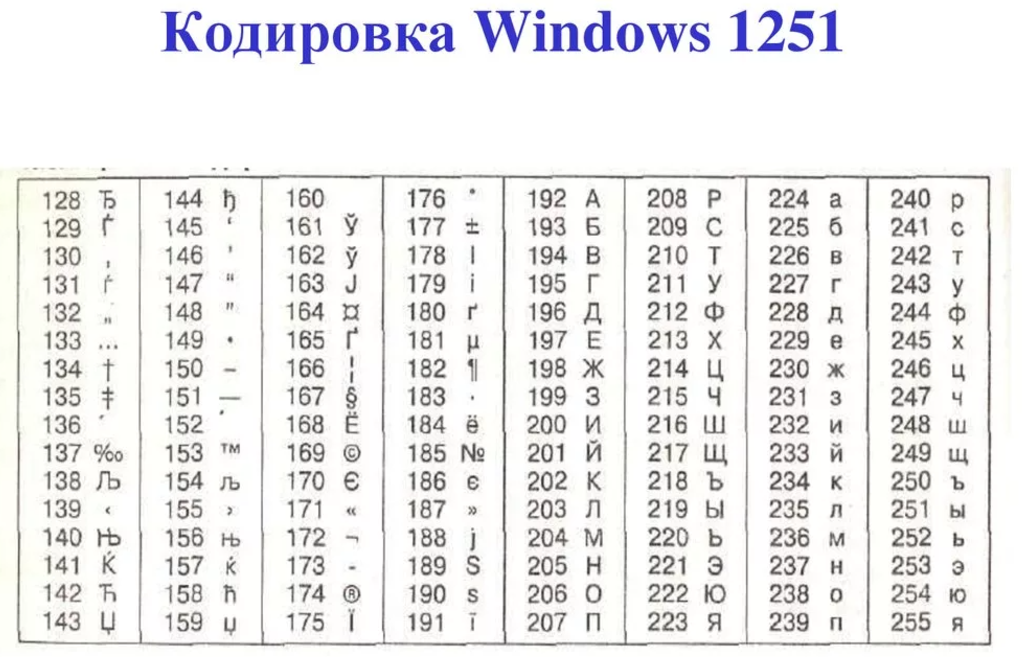
(Ниже приведен нужный фрагмент таблицы)

Из нее получаем, что русская буква «а»= 101000002



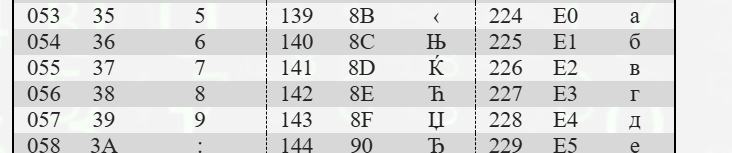
* 1. **Запишите в двоичном и шестнадцатеричном коде в кодировке Windows-1251 русскую букву Я.**

Windows-1251 — набор символов и кодировка, являющаяся стандартной 8-битной кодировкой для русских версий Microsoft Windows до 10-й версии.



Из таблицы получаем, что

* 1. **Запишите символ, который в кодировке Windows-1251 имеет двоичный код 110110.**

****

*=> «6»*

* 1. **В ОС Windows в командном интерпретаторе (*cmd*) изучить команды смены кодовых страниц. Записать текст в кодировке CP866 в отдельный файл. Убедиться, что кодировка текста CP866.**

Команда **CHCP** используется для просмотра или изменения текущей кодовой страницы в окне командной строки Windows. Кодовая страница (Code Page или сокращенно CP) определяет соответствие между двоичным кодом и соответствующим ему символом, отображаемом на экране. Для кодирования текстов на русском языке (то есть букв кириллицы) наиболее широко применяются следующие кодовые страницы:  
  
- Windows-1251, она же Microsoft code page 1251 (CP1251) в операционных системах семейства Windows;  
  
- Кодовая страница CP866, она же IBM code page 866 — в операционных системах семейства DOS, а также в приложениях командной строки Microsoft Windows;  
  
- Кодовая страница CP65001, она же Unicode, UTF-8 — в современных операционных системах семейства Microsoft Windows;  
  
Стандартно, в приложениях командной строки, используется кодовая страница **866**, что соответствует DOS-кодировке. Окно приложения **CMD.EXE** запускается с установленной кодовой страницей **866**. Изменение кодовой страницы действует только по отношению к текущему сеансу CMD.

Вот несколько примеров использования :chcp

*1. Просмотр текущей кодовой страницы:*

***chcp***

Эта команда выведет текущую активную кодовую страницу. Например, указывает на кодовую страницу OEM 866 (кириллица для DOS).Active code page: 866

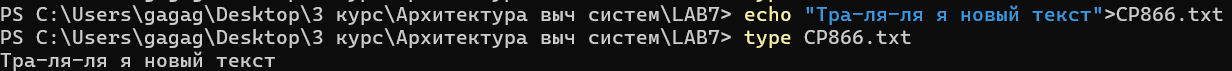
*2. Изменение кодовой страницы на 1251 (кириллица Windows):*

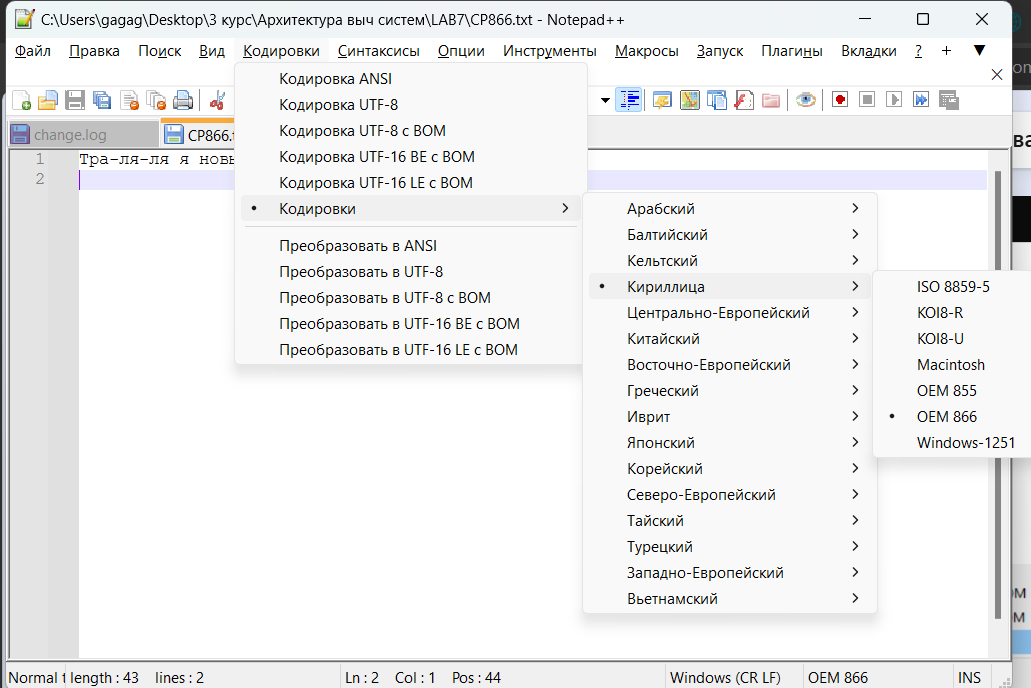
***chcp 1251***

После выполнения этой команды, консоль будет использовать кодовую страницу 1251, которая обычно используется для отображения кириллицы в Windows. Это может быть необходимо, если вы работаете с текстом на русском или других кириллических языках.

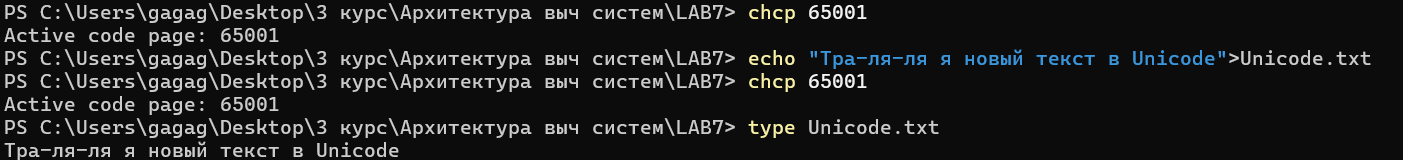
*3. Изменение кодовой страницы на 65001 (UTF-8):*

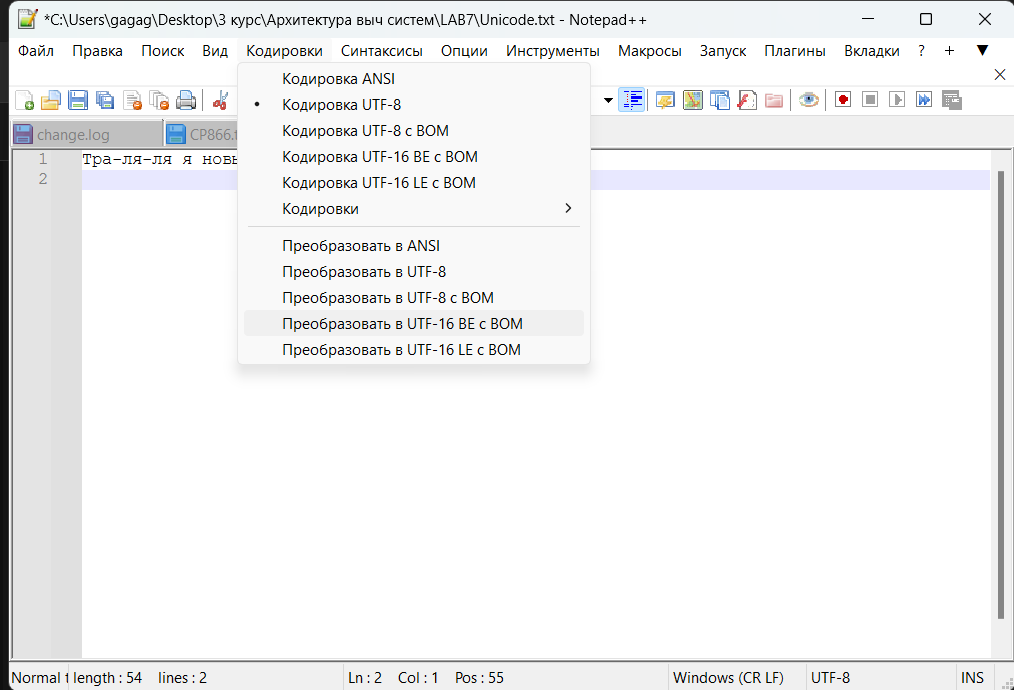
***chcp 65001***

******

******

* 1. В ОС Windows в командном интерпретаторе (***cmd***) изучить команды смены кодовых страниц. Записать текст в кодировке Unicode в отдельный файл. Убедиться, что кодировка текста Unicode.





* 1. В ОС Unix в командном интерпретаторе (***bash***) изучить команды смены кодовых страниц

***$ file -bi [filename]***

*Эта команда определяет тип файла и его кодировку. указывает на краткий вывод, а — на вывод MIME-типа и кодировки.-b-i*

*Пример:*

*file -bi myfile.txt*

*Возможный вывод:*

*text/plain; charset=utf-8*

*Это говорит о том, что — это текстовый файл, закодированный в UTF-8. Эта команда не меняет кодировку, а только определяет её.myfile.txt*

***2. iconv -f [кодировка] -t [кодировка] -o [имя\_нового файла] [имя файла]***

*Команда предназначена для преобразования кодировки*

* + - *-f [encoding] — исходная кодировка.*
* *-t [encoding] — целевая кодировка.*
* *-o [newfilename] — имя нового файла, куда будет записан результат конвертации.*
* *[filename] — имя исходного файла.*

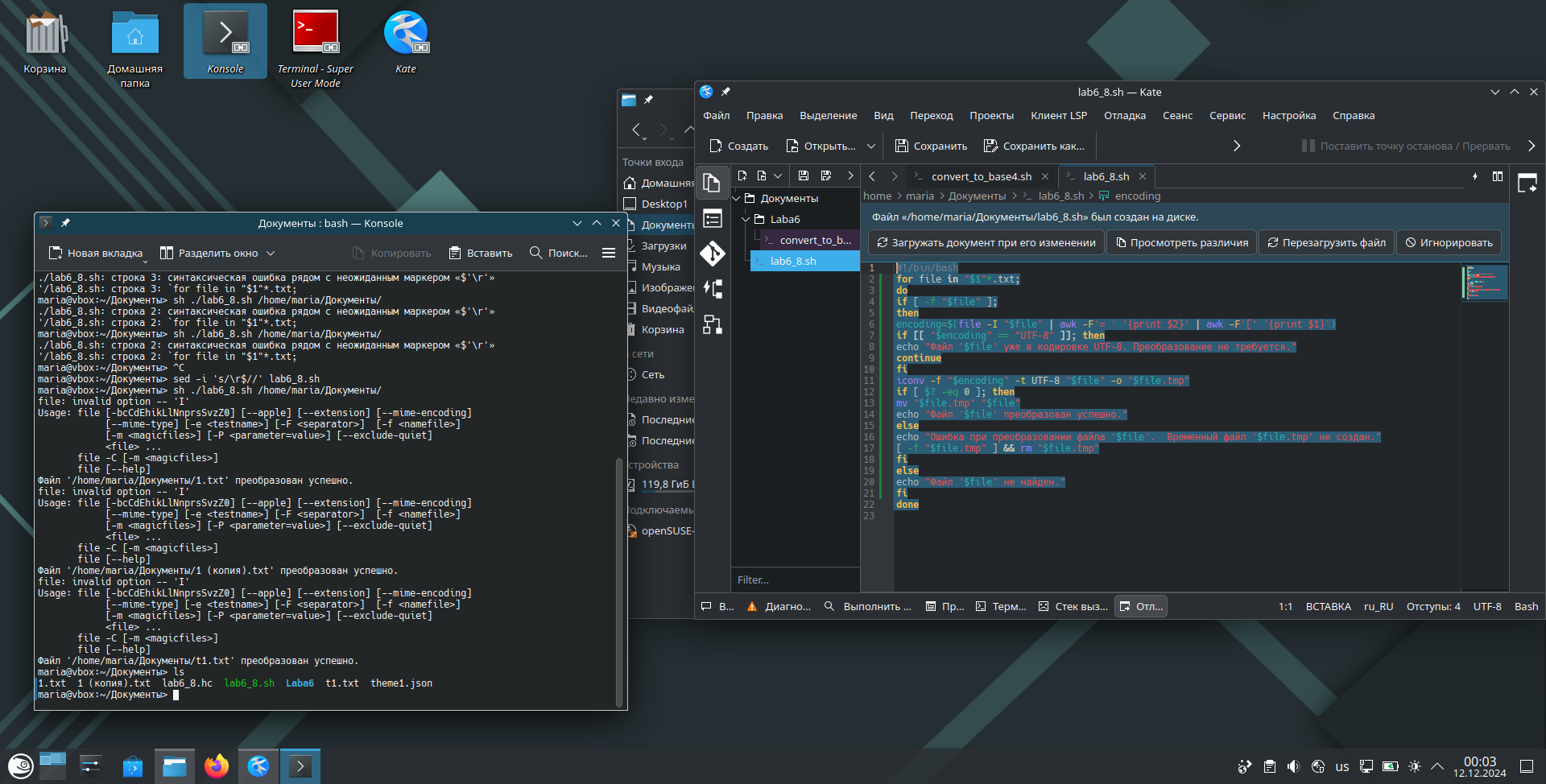
*Пример:*

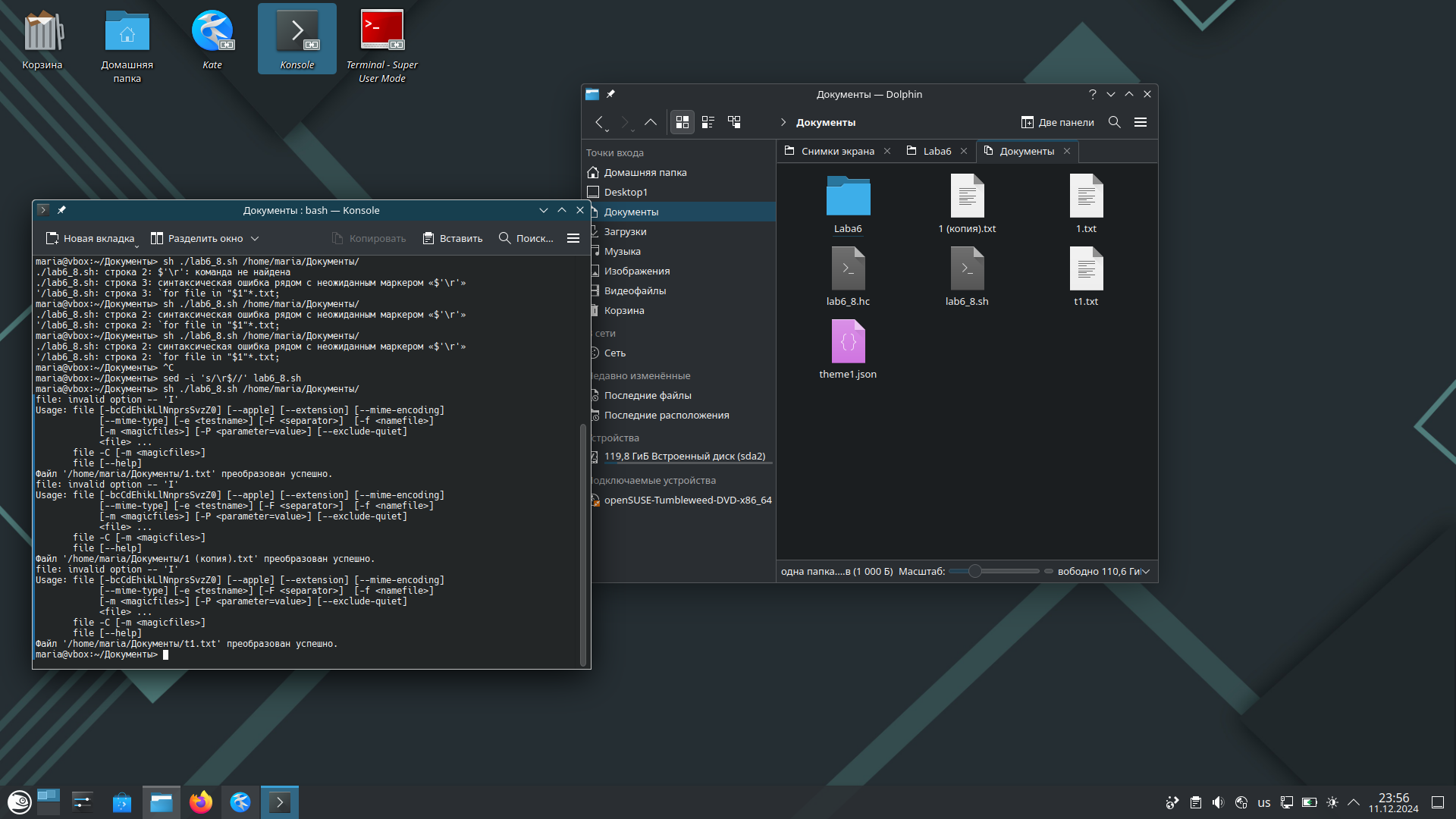
*iconv -f ISO-8859-1 -t UTF-8 input.txt -o output.txt*

* 1. В ОС UNIX изучить команду ***iconv*** для перекодирования текста. Решить задачу: перекодировать содержимое всех текстовых файлов (\*.txt) в заданном каталоге.

Выполнение программы

Структура директории, в которой меняем кодировку файлов .txt





Код программы:

#!/bin/bash

for file in "$1"\*.txt; do

if [ -f "$file" ]; then

# Определяем кодировку файла с помощью file

encoding=$(file -I "$file" | awk -F'= ' '{print $2}' | awk -F'[' '{print $1}')

# Проверяем, является ли кодировка UTF-8

if [[ "$encoding" == "UTF-8" ]]; then

echo "Файл '$file' уже в кодировке UTF-8. Преобразование не требуется."

continue # Переходим к следующему файлу

fi

# Если кодировка не UTF-8, пытаемся преобразовать в UTF-8

iconv -f "$encoding" -t UTF-8 "$file" -o "$file.tmp"

if [ $? -eq 0 ]; then

mv "$file.tmp" "$file"

echo "Файл '$file' преобразован успешно."

else

echo "Ошибка при преобразовании файла '$file'. Временный файл '$file.tmp' не создан."

[ -f "$file.tmp" ] && rm "$file.tmp"

fi

else

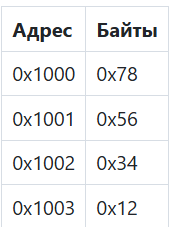
echo "Файл '$file' не найден."

fi

* 1. Следующее задание выполняется по вариантам, приведенным в таблице. Разработайте алгоритм перевода строки символов из «Откуда» в «Куда» с соблюдением указанного «Порядка»:

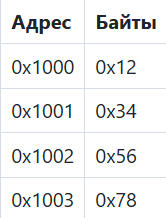
Для начала разберемся в LITTLE\_ENDIAN.

* **Описание:** В LITTLE-ENDIAN порядок байтов обратный: младший байт располагается по самому низкому адресу памяти, а старший — по самому высокому.
* **Пример:** То же 32-битное целое число 0x12345678 в LITTLE-ENDIAN будет представлено как:



Сравним с BIG-ENDIAN

* **Описание:** В BIG-ENDIAN порядок байтов в «многобайтовом» числе такой, что старший байт (байт с наибольшим весом) располагается по самому низкому адресу памяти, а младший байт — по самому высокому.
* **Пример:** Рассмотрим 32-битное целое число 0x12345678. В BIG-ENDIAN оно будет представлено в памяти как:



**Влияние на перекодировку:**

* **8-битные кодировки:** BIG-ENDIAN и LITTLE-ENDIAN не влияют на перекодировку между 8-битными кодировками.
* **16-битные и 32-битные кодировки (UTF-16, UTF-32):** Необходимо указать порядок байтов (BIG-ENDIAN или LITTLE-ENDIAN) при чтении и записи файлов, использующих эти кодировки, чтобы правильно интерпретировать символы. Неправильное указание порядка байтов приведёт к искажению текста.

В общем, при перекодировке важно учитывать *кодировку* (UTF-8, Windows-1251, KOI8-R и т.д.), при этом порядок байтов (BIG-ENDIAN/LITTLE-ENDIAN) становится критичным только при работе с многобайтными кодировками, такими как UTF-16 и UTF-32.

Перейдем к решению задачи:

|  |  |  |  |
| --- | --- | --- | --- |
| № Варианта | Откуда | Куда | Порядок |
| 7 | ANSI | KOI-8R | LITTLE\_ENDIAN |

***Входные данные:***

В папке bin\Debug\net6.0 файл с входными данными <ansi.txt>

***Выходные данные:***

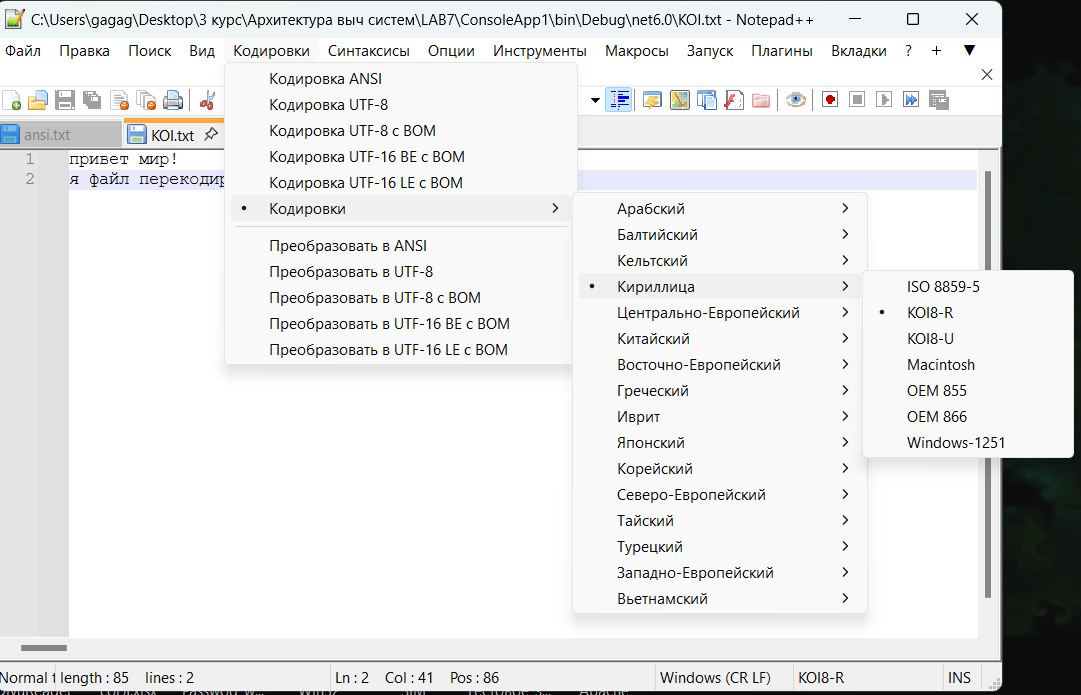
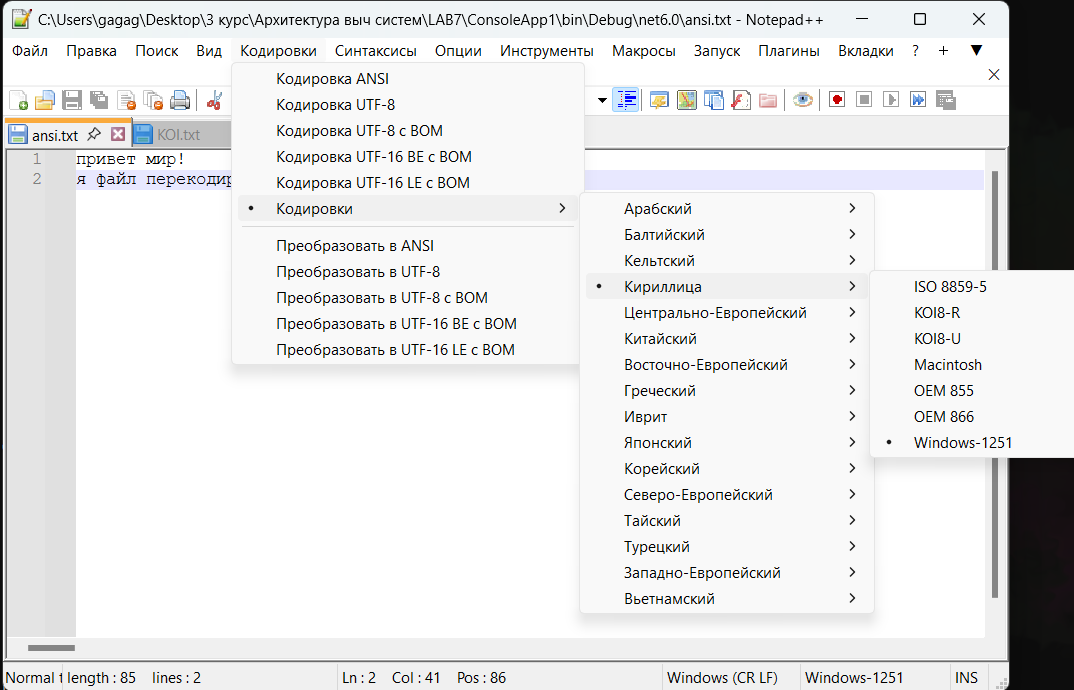
В папке bin\Debug\net6.0 файл с входными данными <KOI.txt>

***Алгоритм.***

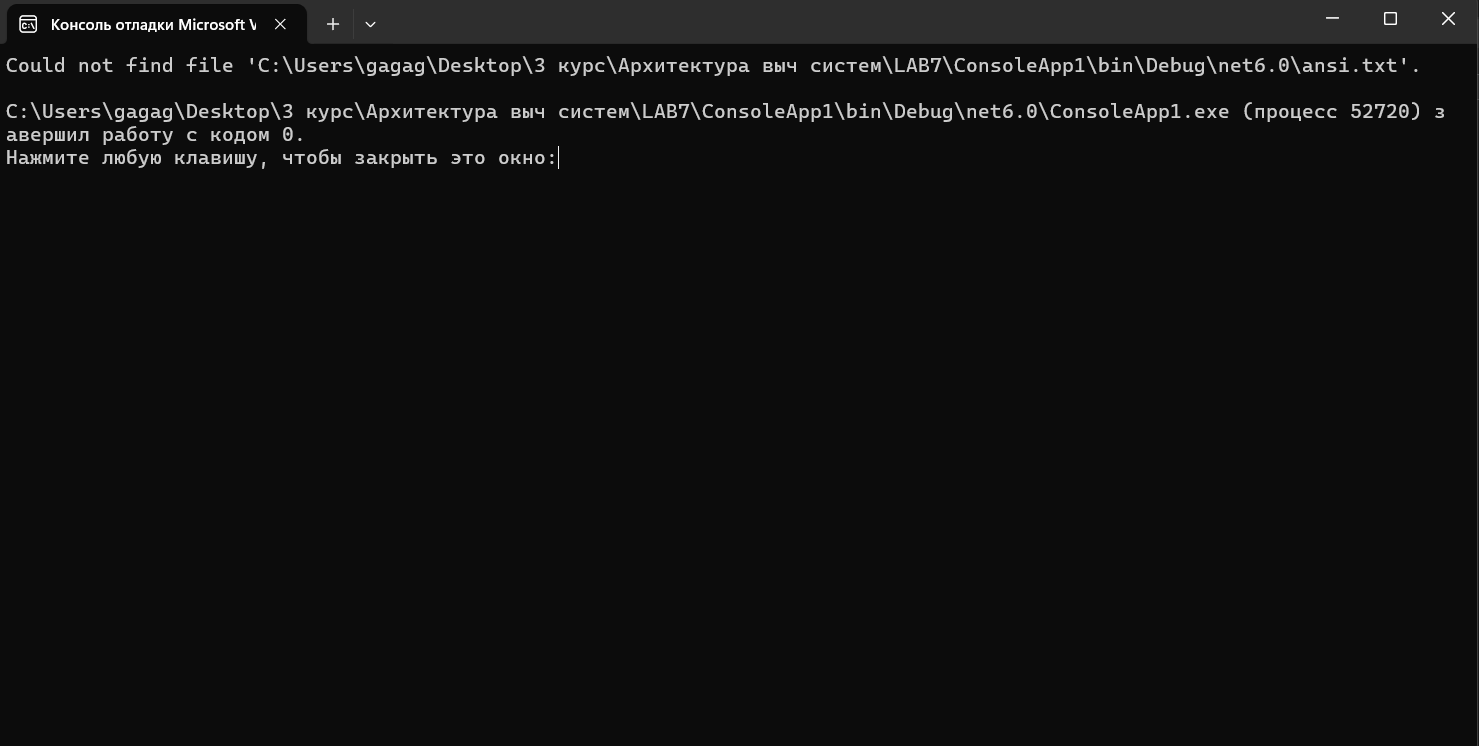
* Считываем файл
* Конвертируем файлы
  + Если нет исключений:
    - Считываем содержимое входного файла (Windows 1251)
    - Создаем будущий выходной файл (KOI-8R)
    - Используем правило для перекодирования порядка LITTLE\_ENDIAN (т.к. перевод между 8-битными кодировками Windows-1251 и KOI8-R, то порядок байтов не меняется, так как каждый символ занимает один байт.)
    - Записываем содержимое входного файла в выходной (outFile.Write(inFile.ReadToEnd());
    - Закрываем файлы.
  + Сообщение в консоль (обработка исключений)

Тестовые примеры:

1.



2.



***Код программы:***

using System;

using System.IO;

using System.Text;

namespace ConvertAnsiToKoi

{

class Program

{

static void Main(string[] args)

{

Encoding.RegisterProvider(CodePagesEncodingProvider.Instance);

string file1, file2;

if (args.Length > 2)

{

file1 = args[0];

file2 = args[1];

}

else

{

file1 = "ansi.txt";

file2 = "KOI.txt";

}

ConvertFile(file1, file2);

}

static void ConvertFile(string inputNameFile, string

outputNameFile)

{

try

{

var inFile = new StreamReader(new FileStream(inputNameFile, FileMode.Open), Encoding.GetEncoding(1251));

var outFile = new StreamWriter(new FileStream(outputNameFile, FileMode.Create), Encoding.GetEncoding(21866));

outFile.Write(inFile.ReadToEnd());

inFile.Close();

outFile.Close();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

}