### Постановка задачи.

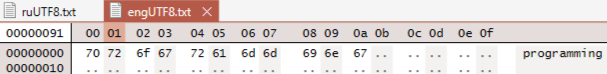
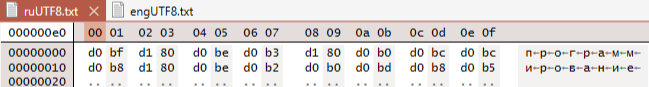
- создать два текстовых файла  
- в одном текст - латиница, в другом - кириллица  
- открыть файлы в HEX-редакторе  
- определить коды символов: латиница, кириллица  
- сверить результат с таблицами ASCII, UNICODE  
- сверить результат с таблицами ASCII, UNICODE в программе на С/С++  
- загрузить файлы в массив байт

### Введение.

Кодировка - это способ представления символов в виде чисел, которые хранятся и обрабатываются компьютером. Кодировка нужна для того, чтобы разные устройства и программы могли обмениваться текстовой информацией без потери или искажения данных.

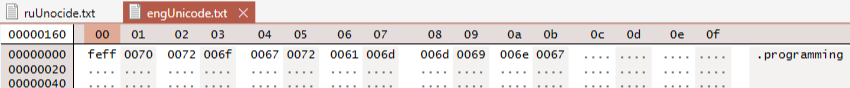
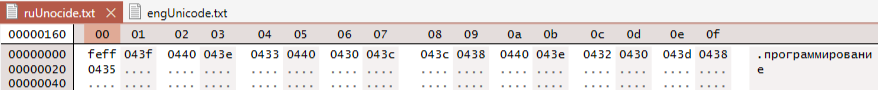
### Решение.

Я создам эти файлы, открою их в HEX-редакторе и определю коды символов в каждом файле. Затем я создам эти файлы в кодировках ASCII и Unicode (с помощью NotePad) и сравню с ними, а также с результатами работы программы на С++, которая загружает файлы в массив байт.

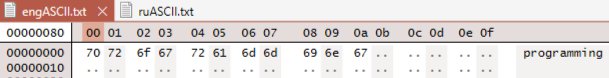
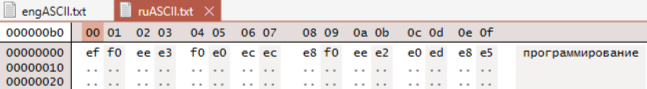


Как мы видим, кодировки разительно отличаются. Теперь сравним файлы других кодировок.

Unicode:



ASCII:



Кодировки латиницы одинаковы. Даже при неправильно выставленной кодировки символы латиницы распознаются правильно (и другие совпадающие по кодам, например, табуляции). Для кириллицы ситуация обратная, более того если в компьютер не включена поддержка кодов кириллицы, даже при выставлении соответствующей кодировки они не распознаются.

Проделаю тоже самое на языке с++.

Эта программа открывает файлы UTF-8.txt, Unicode.txt и ASCII.txt и заносит содержимое каждого файла в отдельный массив байт.

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

int main()

{ // Создаем вектор имен файлов

vector<string> fileNames = { "engUTF-8.txt", "engUnicode.txt", "engASCII.txt","ruUTF-8.txt", "ruUnicode.txt", "ruASCII.txt" };

// Создаем вектор массивов байт

vector<vector<char>> byteArrays;

// Проходим по каждому имени файла в векторе

for (const auto& fileName : fileNames)

{

// Открываем файл в двоичном режиме

//ios::binary - флаг двоичного режима

ifstream file(fileName, ios::binary);

// Получаем размер файла

file.seekg(0, ios::end); //Переместили курсор в конец файла

const auto fileSize = file.tellg(); //Узнали расположение курсора = размер файла

file.seekg(0, ios::beg); //Вернули курсор в начало

// Читаем содержимое файла в массив байт

vector<char> bytes(fileSize);

file.read(bytes.data(), fileSize);

// Добавляем массив байт в вектор массивов байт

byteArrays.push\_back(bytes);

}

//Создаем счетки для отделения RUS и ENG файлов

int counter = 0;

// Проходим по каждому массиву байт в векторе массивов байт

for (const auto& byteArray : byteArrays)

{

if (counter == 0){

cout << "ENG: " << endl;

} else if (counter == 3) {

cout <<endl <<"RUS: " << endl;

}

counter++;

// Проходим по каждому байту в массиве байт

for (const auto& byte : byteArray)

{

// Выводим значение байта в десятичном формате

cout << static\_cast<int>(byte) << " ";

}

// Переходим на новую строку после вывода содержимого массива байт

cout << endl;

}

return 0;

}

### Заключение.

В результате работы я выяснила, что кодировка ASCII использует один байт для представления символов латиницы, Unicode использует два байта для представления символов кириллицы. [А UTF-8 кодирует символы латиницы, которые совпадают с ASCII, одним байтом, а символы кириллицы - двумя или тремя байтами](https://en.wikipedia.org/wiki/UTF-8).

Это объясняет разницу в размерах файлов и в HEX-представлении текста. Также я убедилась, что программа на С++ корректно читает и выводит коды символов из обоих файлов, используя массив байт. Эта работа показала мне важность выбора подходящей кодировки для обработки текстовой информации.