Учреждение образования

Белорусский государственный технологический университет

Кафедра полиграфического оборудования и

системы обработки информации

**Отчет по лабораторной работе**

«Применение стандартов кодирования

текстовой информации»

по дисциплине «Стандартизация и сертификация

информационных систем и технологий»

Выполнил студент

ФИТ ПОИТ 5-2 Валдайцев А. Д.

Проверил

кандидат технических наук

Сулим П.Е.

Отчет по лабораторной работе

защищен с отметкой баллов

Минск 2024

***Цель* *работы*:**

Представление текстовой информации – разобраться со стандартами и ответить на вопросы.

Стандарты для следующих *coded character sets*:

* *ASCII* *character set*
* *ANSI character set*
* *Universal Coded Character Set*

Освоить понятие *Unicode* кодеки: *UTF*-8, *UTF*-16, *UTF*-32.

Освоить правила записей кодов символов.

Привести примеры записей кодов символов

Разобраться в понятиях *code point*, *code unit*, *character set*, *coded character set*.

Привести примеры кодирования текстовой информации.

Приведите краткие выводы по содержанию работы.

## Coded Character Set

1. *ASCII* *character set (7 bits)*

***A****merican****s****tandard****c****ode for****i****nformation****i****nterchange*.

**ASCII** (American Standard Code for Information Interchange) — это первая кодировка, которая была использована в компьютерах с 1963 года. Она представляет собой 7-битную кодировку, что означает, что в ней определено 128 символов. Восьмой бит полного байта использовался для проверки четности, так как в то время каналы связи были ненадежными, и предполагалось, что каждый полученный байт будет проверяться.

1. *ANSI* *character set (8 bits = 1 byte)*

***A****merican****n****ational****s****tandards****i****nstitute*

Со временем стало очевидно, что восьмой бит можно использовать для отображения национальных символов других языков, что позволит использовать 256 символов. Это расширенное 8-битное кодирование условно называется **ANSI** (American National Standards Institute) по имени американского института стандартов, который предложил это 8-битное кодирование. Соответственно, для каждого национального языка была предложена своя раскладка символов во второй половине таблицы (от 128 до 255), а первая половина таблицы от 0 до 127 содержала исходные символы ASCII.

1. *Universal Coded Character Set (Unicode) (2 bytes)*

Когда возникла потребность в представлении широкого спектра символов, включая иероглифы, стало очевидно, что ограниченный набор из 256 символов уже недостаточен. В результате был разработан и внедрен стандарт **Unicode**, который предоставляет механизм для кодирования символов из различных письменностей и языков. В стандарте Unicode каждому символу выделяется 2 байта, позволяющих представить 65 536 различных символов.

1. *UTF (Unicode Transformation Format) (from 1 byte to 4 bytes)*

Однако с течением времени стало очевидно, что даже в Unicode недостаточно места для представления всех символов, так как появились новые символы, включая эмодзи и специфичные национальные символы. В ответ на эту проблему был разработан и предложен стандарт **UTF-8 (Unicode Transformation Format 8-bit)**, который представляет символы Unicode с использованием переменной длины кодирования. В UTF-8 количество байтов, необходимых для представления символа, может быть от 1 до 4 байт, что позволяет представить более 1 миллиона символов. UTF-8 является одним из наиболее широко используемых методов кодирования символов Unicode и обеспечивает совместимость с ASCII для символов.

## Unicode кодеки

1. *UTF-8*

UTF-8 обладает преимуществом в случае, когда ASCII-символы составляют большую часть символов в текстовом блоке, поскольку UTF-8 кодирует их в 8 бит (как ASCII). Еще одним преимуществом является то, что файл, закодированный в UTF-8 и содержащий только ASCII-символы, имеет ту же кодировку, что и файл в ASCII.

1. *UTF-16*

UTF-16 лучше подходит в случае, когда ASCII не является преобладающим, поскольку он использует 2 байта на символ в основном. В то время как UTF-8 начинает использовать 3 или более байта для символов с более высоким порядком, UTF-16 сохраняет размер в 2 байта для большинства символов.

1. *UTF-32*

UTF-32 позволяет представить все возможные символы в 4 байтах. Это делает его достаточно громоздким. Кодировка специфична и применяется редко. Главное преимущество UTF-32 перед кодировками переменной длины заключается в том, что символы Юникод непосредственно индексируемы. Получение n-ой кодовой позиции является операцией, занимающей одинаковое время. Напротив, коды с переменной длиной требует последовательного доступа к n-ой кодовой позиции. Это делает замену символов в строках UTF-32 простой, для этого используется целое число в качестве индекса, как обычно делается для строк ASCII.

1. *LE, BE*

От этого зависит порядок бит в байте – слева направо (**Big Endian**) или справа налево (**Little Endian**). Например, 0x12345678 в LE это 78 56 34 12, а в BE – 12 34 56 78. LE и BE могут использоваться вместе с UTF кодировкой (8, 16 или 32).

## Примеры записей кодов символов

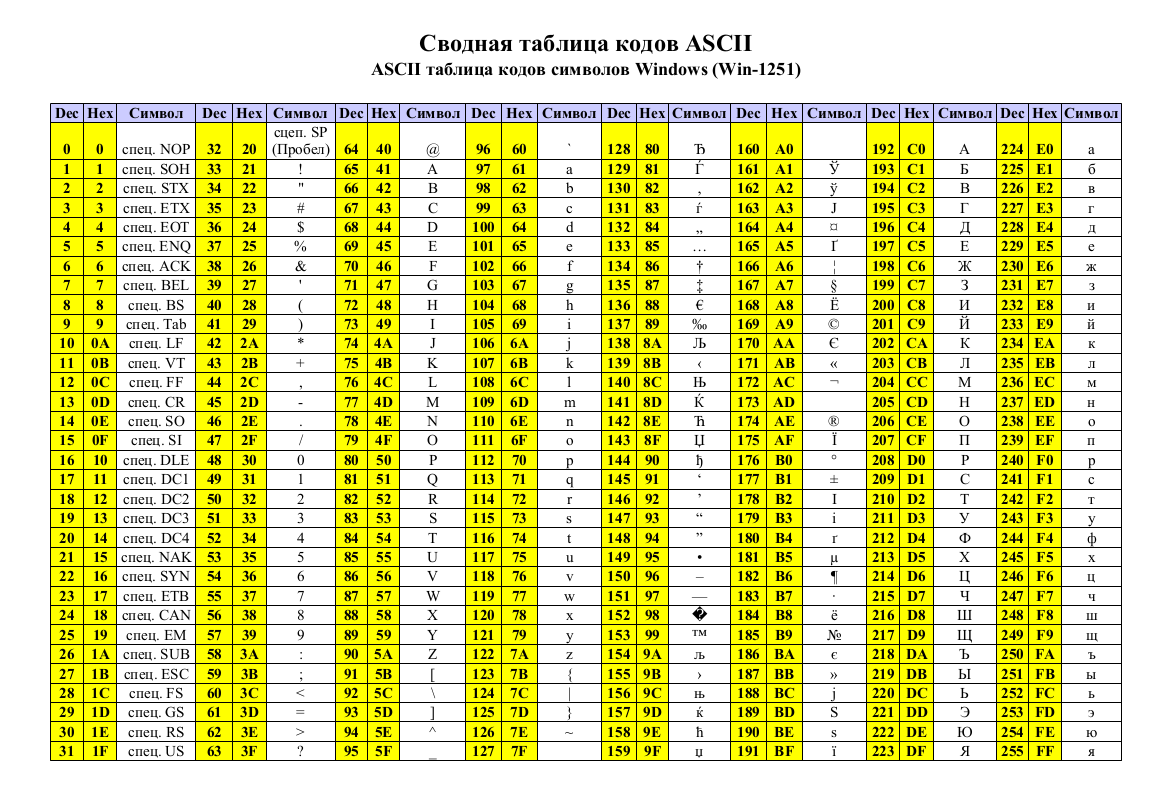


Рисунок 1.1 – ASCII (расширенная)

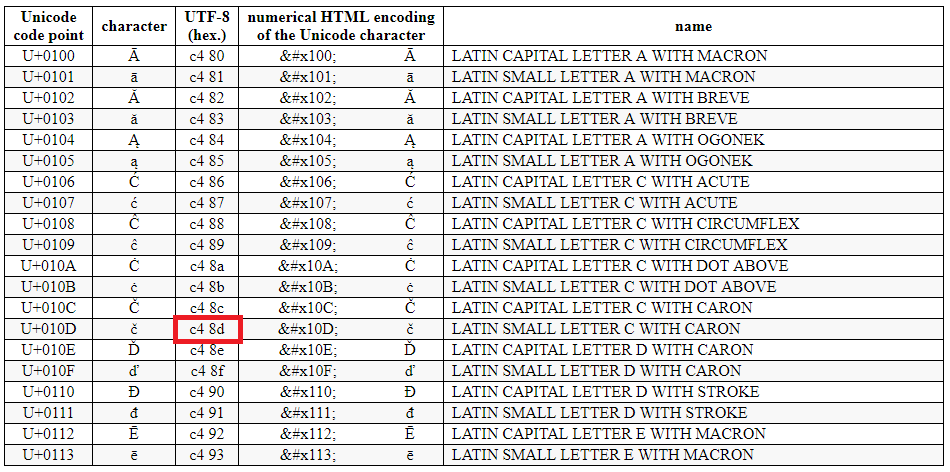


Рисунок 1.2 – UTF-8 и Unicode Code Point



Рисунок 1.3 – UTF-8, CP-1251 и ISO-8859-1

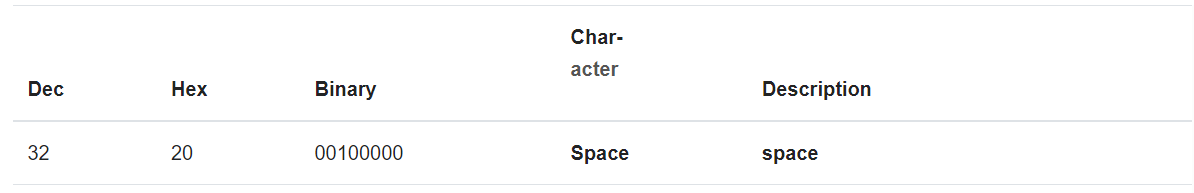


Рисунок 1.4 – Dec, Hex, Binary в ASCII

## Понятия

*Code point* – представляет собой числовое значение, которое соответствует отдельному символу в определенной кодировке. Например, символ "A" в кодировке ASCII имеет свой уникальный code point, который равен 65.

*Code unit* – это минимальная единица информации в определенной кодировке. В различных кодировках code unit может иметь различную длину. Например, в кодировке ASCII, каждый символ представляется одним code unit (8 бит), а в кодировке UTF-8, code unit может иметь длину от 8 до 32 бит в зависимости от символа.

*Character set* – представляет собой определенный набор символов, которые могут быть использованы в тексте. Например, ASCII character set состоит из основных латинских букв, цифр и специальных символов. Unicode character set является более широким набором, который включает символы различных письменностей и языков.

*Coded character set* – это character set, который был присвоен определенной кодировке. Он представляет собой соответствие между символами из character set и их числовыми code point в данной кодировке. Например, в кодировке UTF-8, Unicode character set закодирован с использованием определенных последовательностей байтов, которые представляют соответствующие code point.

## Алгоритм кодирования в UTF-8

1. Определить количество октетов (байтов), требуемых для кодирования символа. Номер символа берётся из стандарта Юникода.

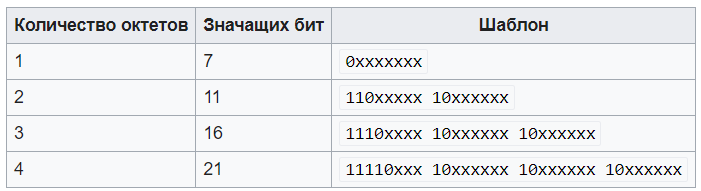


Для символов Юникода с номерами от U+0000 до U+007F (занимающими один байт c нулём в старшем бите) кодировка UTF-8 полностью соответствует кодировке ASCII.

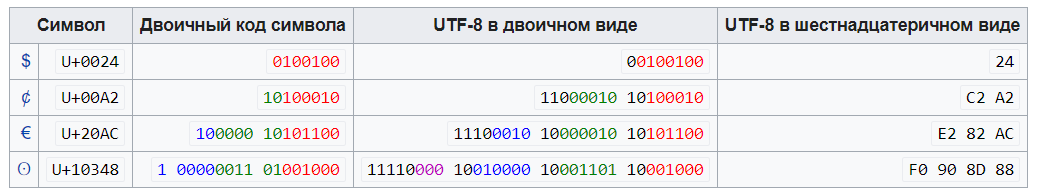
1. Установить старшие биты первого октета в соответствии с необходимым количеством октетов, определённом на первом этапе:

* 0xxxxxxx — если для кодирования потребуется *один* октет;
* 110xxxxx — если для кодирования потребуется *два* октета;
* 1110xxxx — если для кодирования потребуется *три* октета;
* 11110xxx — если для кодирования потребуется *четыре* октета.

Если для кодирования требуется больше одного октета, то в октетах 2-4 два старших бита всегда устанавливаются равными 102 (10xxxxxx). Это позволяет легко отличать первый октет в потоке, потому что его старшие биты никогда не равны 102.



1. Установить значащие биты октетов в соответствии с номером символа Юникода, выраженном в двоичном виде. Начать заполнение с младших битов номера символа, поставив их в младшие биты последнего октета, продолжить справа налево до первого октета. Свободные биты первого октета, оставшиеся незадействованными, заполнить нулями.
2. Примеры:



## Пример кодирования

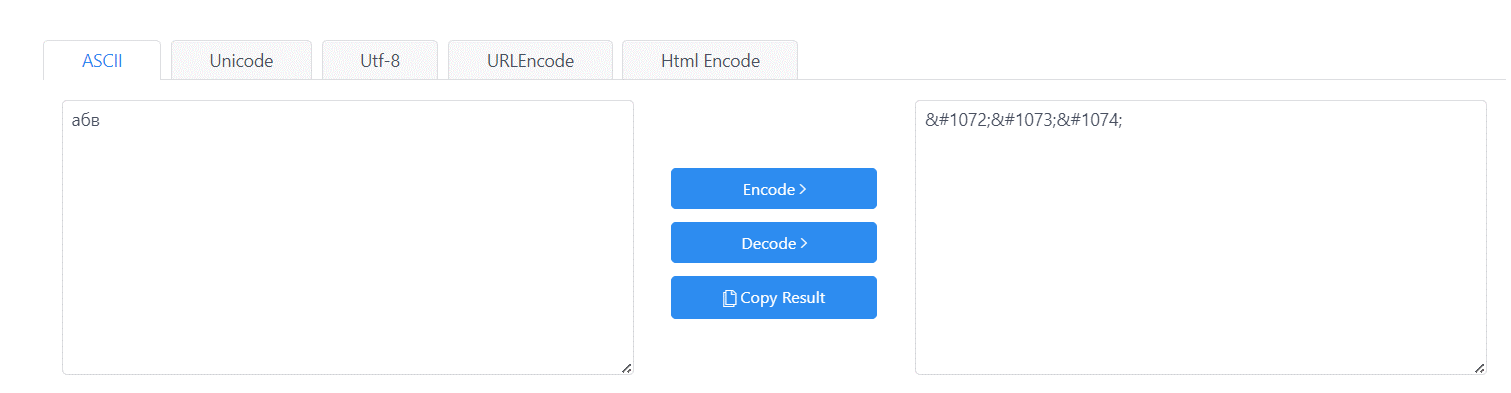


Рисунок 1.5 – ASCII

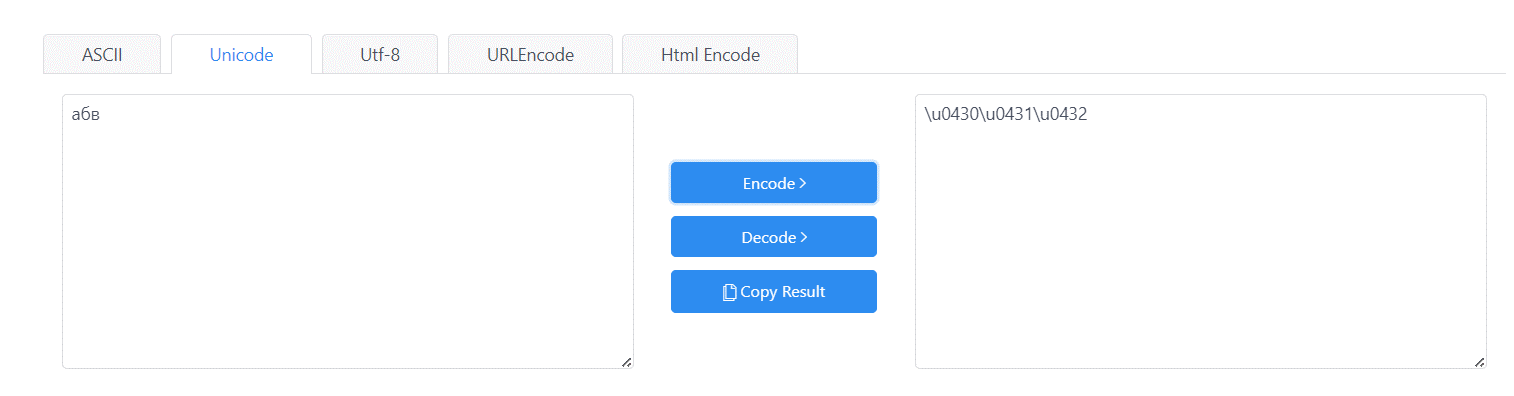


Рисунок 1.6 – Unicode

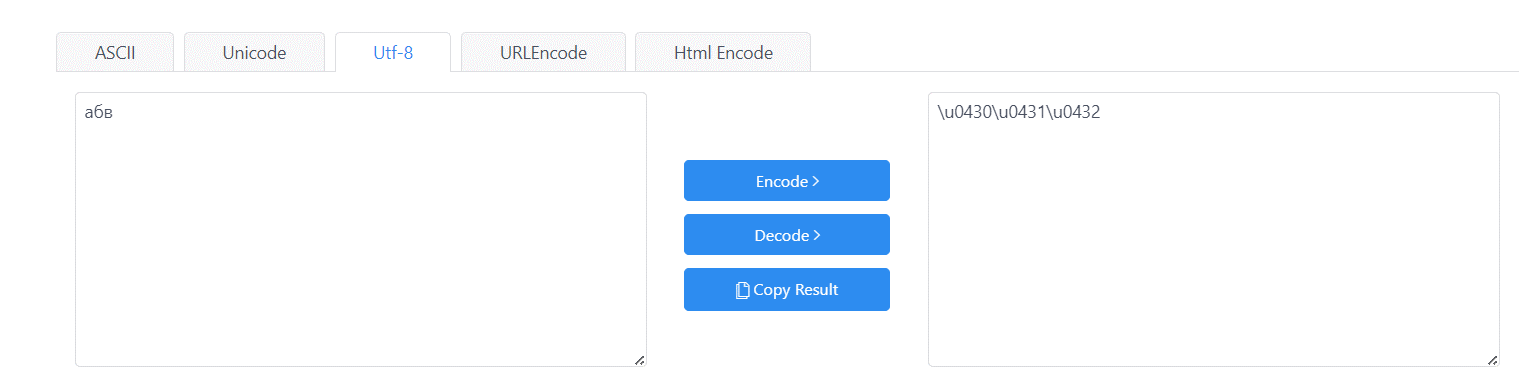


Рисунок 1.7 – UTF-8

## Источники

[Что есть ANSI и ASCII - Stack Overflow на русском](https://ru.stackoverflow.com/questions/872902/%D0%A7%D1%82%D0%BE-%D0%B5%D1%81%D1%82%D1%8C-ansi-%D0%B8-ascii?ysclid=lsyx8rtdqo786306496)

[UTF-8, UTF-16, and UTF-32 - Stack Overflow](https://stackoverflow.com/questions/496321/utf-8-utf-16-and-utf-32)

[What Is Little-Endian And Big-Endian Byte Ordering? | by Pragma08 | Medium](https://captainlink.medium.com/what-is-little-endian-and-big-endian-byte-ordering-3768c065f45e)

[ASCII — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/ASCII)

[Американский национальный институт стандартов — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/%D0%90%D0%BC%D0%B5%D1%80%D0%B8%D0%BA%D0%B0%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D0%BD%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%81%D1%82%D0%B8%D1%82%D1%83%D1%82_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BE%D0%B2)

[Windows-1251 — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/Windows-1251)

[Universal Coded Character Set - Wikipedia](https://en.wikipedia.org/wiki/Universal_Coded_Character_Set)

[ASCII Table | Tachyon (grandidierite.github.io)](https://grandidierite.github.io/ASCII-table/)

[Виды кодировок символов (dieg.info)](https://wiki.dieg.info/kodirovka)

## Выводы

В данной лабораторной работе были рассмотрены различные стандарты и понятия, связанные с представлением текстовой информации – ASCII, Unicode, UTF. Существует множество стандартов и кодировок для представления текстовой информации, и выбор конкретной зависит от требований и особенностей конкретного приложения.

## Дополнительный вопрос

Данный символ (Ö) получается с помощью клавиш клавиуатуры ALT+0214.