## Учреждение образования

## Белорусский государственный технологический университет

Кафедра полиграфического оборудования и

системы обработки информации

**ЛАБОРАТОРНАЯ РАБОТА № 3**

по дисциплине «Стандартизация и сертификация

информационных систем и технологий»

Тема

«Применение стандартов кодирования

текстовой информации»

Выполнил студент:

Коржова В. С.

4 курс 4 группа

Проверил

кандидат технических наук

Сулим П.Е.

Отчет по лабораторной работе

защищен с отметкой баллов

Минск 2024

Лабораторная работа № 3

**ПРИМЕНЕНИЕ СТАНДАРТОВ КОДИРОВАНИЯ**

**ТЕКСТОВОЙ ИНФОРМАЦИИ**

***Цель* *работы*:** Представление текстовой информации – разобраться со стандартами и ответить на вопросы.

***Теоретические сведения:***

**Кодовая точка - Code point** – числовое значение, представляющее символ в наборе кодированных символов

В терминологии кодировки символов, кодовая точка или позиция кода представляет собой любое числовое значение, составляющее пространство кода. Многие кодовые точки представляют собой отдельные символы, но они также могут иметь другие значения, например, для форматирования.

**Code unit** – это единицы кодировки. Байт для utf-8, Слово (два байта) для utf-16 или Длинное слово (четыре байта) для utf-32.

**Набо́р си́мволов (англ. character set)** – таблица, задающая кодировку конечного множества символов алфавита (обычно элементов текста: букв, цифр, знаков препинания). Такая таблица сопоставляет каждому символу последовательность длиной в один или несколько символов другого алфавита (точек и тире в коде Mорзе, сигнальных флагов на флоте, нулей и единиц (битов) в компьютере).

**UCS (англ. Universal Coded Character Set)** представляет собой стандартный набор символов, определенный международным стандартом ISO/IEC 10646, который является основой многих символьных кодировок. UCS содержит чуть более 128 000 абстрактных символов, как и в Unicode 9.0, каждый из которых определяется однозначно сочетанием имени и целого числа (так называемый "кодовый пункт"). Символы (буквы, цифры, идеограммы, логограммы и т.д.) из многих языков, манускриптов, традиций мира представлены в UCS с уникальными кодовыми пунктами. Символы из ранее не представленных письменностей добавляются в UCS, постоянно совершенствуя ее.

Unicode определяет один огромный набор символов, присваивая каждому графическому символу одно уникальное целочисленное значение. UTF-8/16/32 – это просто разные способы кодирования этого.

UTF-32 использует 32-битные значения для каждого символа. Это позволяет им использовать код фиксированной ширины для каждого символа.

UTF-16 по умолчанию использует 16-битные символы, но это дает вам только 65 тысяч возможных символов, что далеко не достаточно для полного набора Unicode. Поэтому некоторые символы используют пары 16-битных значений.

А UTF-8 по умолчанию использует 8-битные значения, что означает, что 127 первых значений являются однобайтовыми символами фиксированной ширины (наиболее значимый бит используется для обозначения того, что это начало многобайтовой последовательности, оставляя 7 бит для фактического значения символа). Все остальные символы кодируются в виде последовательностей длиной до 6 байт.

И это приводит нас к преимуществам. Любой символ ASCII напрямую совместим с UTF-8, поэтому для обновления устаревших приложений UTF-8 является обычным и очевидным выбором. Почти во всех случаях он также будет использовать наименьшее количество памяти. С другой стороны, вы не можете дать никаких гарантий относительно ширины символа. Он может быть шириной 1, 2, 3 или 4 символа, что затрудняет манипуляцию строками.

Кодировка UTF-8 сейчас является доминирующей в веб-пространстве. Она также нашла широкое применение в UNIX-подобных операционных системах.

UTF-32 – это наоборот, он использует большую часть памяти (каждый символ имеет фиксированную ширину 4 байта), но с другой стороны, вы знаете, что каждый символ имеет именно такую длину, поэтому манипуляция строками становится намного проще. Вы можете вычислить количество символов в строке просто из длины строки в байтах. Вы не можете сделать это с UTF-8.

UTF-16 – это компромисс. Он позволяет большинству символов вписываться в 16-битное значение фиксированной ширины. Поэтому, пока у вас нет китайских символов, музыкальных нот или каких-то других, вы можете считать, что каждый символ имеет ширину 16 бит. Он использует меньше памяти, чем UTF-32. Он почти всегда использует больше памяти, чем UTF-8, и все равно не избегает проблемы, которая преследует UTF-8 (символы переменной длины).

Все три кодировки могут кодировать один и тот же набор символов, но они представляют каждый символ в виде различных последовательностей байтов.

***ASCII*** — **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange.

ASCII была разработана (1963 год) для кодирования символов, коды которых помещались в 7 бит (128 символов). Со временем кодировка была расширена до 8-ми бит (256 символов), коды первых 128-и символов не изменились.

***ANSI*** – **A**merican **N**ational **S**tandard **I**nstitute.

Каждый символ кодируется однобайтным кодом. Коды от 0 до 127 совпадают с кодами ASCII, коды от 128 до 255 могут означать разные символы различных языков в зависимости от выбранной кодовой страницы.

ANSI-кодировка – это собирательное название. В действительности, реальная кодировка при использовании ANSI будет определяться тем, что указано в реестре вашей операционной системы Windows. В случае с русским языком это будет Windows-1251, однако, для других языков это будет другая разновидность ANSI.

***Unicode*** (***1991***)— стандарт кодирования символов, включающий в себя знаки почти всех письменных языков мира. В настоящее время стандарт является преобладающим в Интернете.

Стандарт состоит из двух основных частей: универсального набора символов (англ. Universal character set, UCS) и семейства кодировок (англ. Unicode transformation format, UTF). Универсальный набор символов перечисляет допустимые по стандарту Юникод символы и присваивает каждому символу код в виде неотрицательного целого числа, записываемого обычно в шестнадцатеричной форме с префиксом U+, например, U+040F. Семейство кодировок определяет способы преобразования кодов символов для передачи в потоке или в файле.

В кодировке Unicode используется от 1 до 6 байт на символ. Этого хватает для символов латиницы и кириллицы, греческого алфавита, китайских иероглифов, арабских и еврейских букв, а также многочисленных дополнительных (финансовых, математических и т. п.) символов. Кодовых страниц в Unicode нет.

**Практическая часть:**

Правила записи кода одного символа в UTF-8

1. Если размер символа в кодировке UTF-8 = 1 байт Код имеет вид

(0aaa aaaa), где «0» — просто ноль, остальные биты «a» — это код символа в кодировке ASCII;

2. Если размер символа в кодировке в UTF-8 >1 байт (то есть от 2 до

6):

2.1 Первый байт содержит количество байт символа, закодированное в

единичной системе счисления;

2 — 11

3 — 111

4 — 1111

5 — 1111 1

6 — 1111 11

2.2 «0» — бит терминатор, означающий завершение кода размера

2.3 далее идут значащие байты кода, которые имеют вид (10xx xxxx), где «10» — биты признака продолжения, а «x» — значащие биты. В общем случае варианты представления одного символа в кодировке UTF-8 выглядят так:

(1 байт) 0aaa aaaa

(2 байта) 110x xxxx 10xx xxxx

(3 байта) 1110 xxxx 10xx xxxx 10xx xxxx

(4 байта) 1111 0xxx 10xx xxxx 10xx xxxx 10xx xxxx

(5 байт) 1111 10xx 10xx xxxx 10xx xxxx 10xx xxxx 10xx xxxx

(6 байт) 1111 110x 10xx xxxx 10xx xxxx 10xx xxxx 10xx xxxx 10xx xxxx

Для обозначения символов Unicode используется запись вида «U+xxxx» (для кодов 0…FFFF), или «U+xxxxx» (для кодов 10000…FFFFF), или «U+xxxxxx» (для кодов 100000…10FFFF), где xxx — шестнадцатеричные цифры.

На рисунке 1 представлены некоторые буквы русского алфавита в Unicоde.



Рисунок 1 – Некоторые символы русского алфавита в Unicode

На рисунке 2 представлены некоторые символы латинского алфавита в кодировке ASCII.

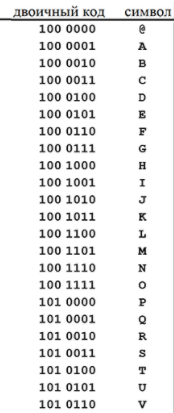


Рисунок 2 – Некоторые символы латинского алфавита в ASCII

На рисунке 3 представлены некоторые символы русского алфавита в кодировке ANSI.

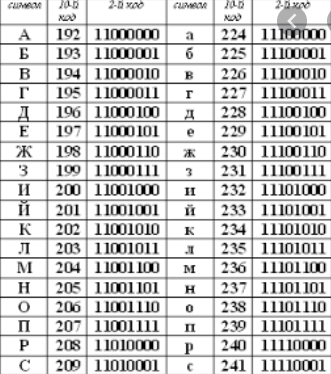


Рисунок 3 – Некоторые символы русского алфавита в ANSI

Слово «Hello» в Unicode -> U-0048 U-0065 U-006C U-006C U-006F

Слово «Hello» в ASCII -> 01001000 01100101 01101100 01101100 01101111

Слово «Hello» в ANSI -> 01001000 01100101 01101100 01101100 01101111

**Вывод:** В этой лабораторной работе были изучены ключевые стандарты кодирования информации, включая ASCII, ANSI и Unicode. Я ознакомилась с различными Unicode кодировками: UTF-8, UTF-16, UTF-32 и освоила методы представления символьных кодов. Также я разобралась в таких понятиях, как code point, code unit, character set и coded character set.