**Тема 4. Измерение объёма информации**

# Измерение объёма текстовой информации

# Измерение объёма графической информации

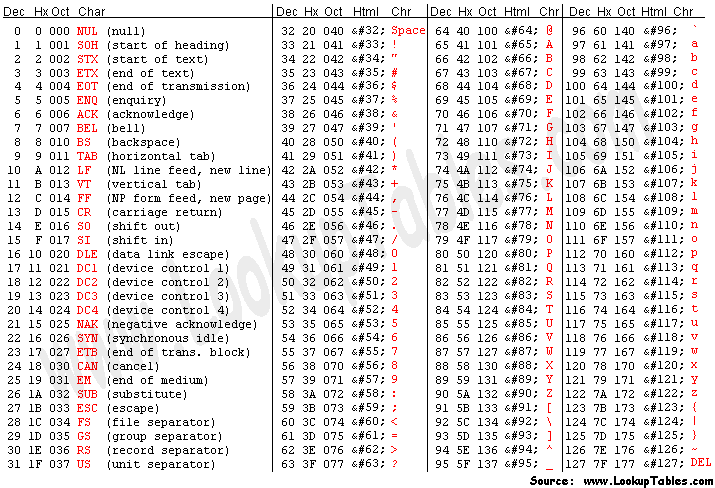
# Измерение объёма текстовой информации

Для того, чтобы измерить информацию, её нужно закодировать. Для кодирования символов нужно определиться с используемым алфавитом, то есть какие именно символы мы будем использовать в тексте, затем определить мощность этого алфавита, то есть количество символов в нём и, наконец, подобрать минимальную разрядность кода с помощью формулы Хартли. Однако, при таком подходе может существовать бесконечное множество различных алфавитов и, соответственно, бесконечное множество различных кодировок. Это неизбежно приведет к путанице. Поэтому кодировки необходимо стандартизировать, привести к единому общему для всех виду.

### **Кодировка ASCII**

Одним из первых распространённых стандартов кодирования был американский стандарт ASCII, разработанный в 1963 году. Изначально в этой кодировке для одного символа отводилось 7 бит информации, этого было достаточно для кодирования десятичных цифр, букв латинского алфавита в верхнем и нижнем регистре, знаков препинания и управляющих символов. Впоследствии её увеличили до 8 бит.





Первые 32 символа – это управляющие символы, такие как backspace, знак табуляции, перенос строки и другие, эти символы визуально не изображаются. Затем с 32 по 64 символ идут знаки препинания, в том числе пробел, и десятичные цифры. Символы с 65 по 90-ый – это заглавные буквы латинского алфавита, а с 97 по 122 – строчные буквы латинского алфавита. Оставшиеся позиции занимают знаки препинания.

К концу 80-ых годов уже существовало большое количество 8-битных кодировок и это породило ряд проблем, связанных с их совместимостью:

**Во-первых,** если в тексте встречались иностранные символы, они могли отображаться некорректно.

**Во-вторых**, зачастую не удавалось вместить все необходимые символы в 1 байт, 256 кодов оказывалось недостаточно.

**В-третьих**, для смены кодировки требовались таблицы перекодировки или промежуточные кодировки. И, наконец, один и тот же шрифт нужно было создавать заново для каждой кодировки, так как коды одних и тех же символов могли отличаться.

***Таким образом, не было ни единого шанса надежно обмениваться данными***.

Интернет — всего лишь огромная сеть компьютеров по всему миру. Представьте, что было бы, если бы каждая страна самостоятельно определяла стандарты кодировки. Тогда компьютеры в Греции нормально выводили бы только греческий язык, а в США — только английский.

**ASCII не подходила для использования в реальном мире. Чтобы интернет был всемирным, нам нужно было что-то менять. Ну, или вечно работать с сотнями кодовых страниц.**

# *И тут пришел Юникод……Абсолютно каждый разработчик программного обеспечения должен знать о Юникоде и наборах символов (без оправданий!)*

### **Кодировка Unicode**

Решением этих проблем стал переход к двухбайтовым кодировкам. В 1991 году появился стандарт кодирования Unicode  (Юникод), в котором изначально для кодирования одного символа отводилось 16 бит. В современных версиях кодировки длина кода может быть переменной и занимать до 32 бит. Это позволило расширить диапазон кодов более чем до миллиона различных значений. Причём на данный момент использованы меньше 150 тысяч кодов, всего 13 процентов от общего объёма.

**Юникод** включает практически все современные письменности, многие исторические письменности, широкий набор математических и музыкальных символов, а также пиктограмм.



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.

Первая версия Юникода представляла собой кодировку с фиксированным размером символа в 16 бит, то есть общее число кодов было 216 (65 536). С тех пор символы стали обозначать четырьмя шестнадцатеричными цифрами (например, U+04F0). При этом в Юникоде планировалось кодировать не все существующие символы, а только те, которые необходимы в повседневном обиходе. Редко используемые символы должны были размещаться в области пользовательских символов, которая первоначально занимала коды U+D800…U+F8FF. Чтобы использовать Юникод также и в качестве промежуточного звена при преобразовании разных кодировок друг в друга, в него включили все символы, представленные во всех наиболее известных кодировках.

Поскольку в ряде компьютерных систем 16-битные символы уже использовались в качестве кодировки по умолчанию, было решено все наиболее важные знаки кодировать только в пределах первых 65 536 позиций. Остальное пространство используется для дополнительных символов: систем письма вымерших языков или очень редко используемых китайских иероглифов, математических и музыкальных символов. Юникод состоит из множества кодовых пунктов (code points, по сути — шестнадцатеричные числа), связанных с символами. Коллекция кодовых пунктов называется набором символов. Вот этот набор — и есть Юникод.

Люди проделали огромную работу, назначив коды всем символам всех языков, а мы можем просто пользоваться результатами их труда. Отображение кодов выглядит так:

"Hello World"

U+0048 : LATIN CAPITAL LETTER H

U+0065 : LATIN SMALL LETTER E

U+006C : LATIN SMALL LETTER L

U+006C : LATIN SMALL LETTER L

U+006F : LATIN SMALL LETTER O

U+0020 : SPACE [SP]

U+0057 : LATIN CAPITAL LETTER W

U+006F : LATIN SMALL LETTER O

U+0072 : LATIN SMALL LETTER R

U+006C : LATIN SMALL LETTER L

U+0064 : LATIN SMALL LETTER D

*U+ указывает на то, что это стандарт Unicode, а номер — число, в которое переведен двоичный код для данного символа. В Юникоде используется шестнадцатеричная система — просто потому, что в нее проще переводить двоичный код. Впрочем, вам не придется делать это вручную, так что можно не волноваться.*

### **Информационный объём текста**

Определившись с кодировкой, вы можете вычислить объём текста по формуле:

I =*k*⋅*i*

k - количество символов в тексте

i - информационный объём одного символа

В зависимости от задачи может использоваться как одна из стандартных кодировок, так и произвольная. Во втором случае может также потребоваться формула Хартли.

Рассмотрим примеры задач на вычисление информационного объёма текста.

**Пример 1.**Определите информационный объём сообщения

Здравствуй, мир!

записанного в 16-битной кодировке.

*Решение:*

*k = 16 символов (13 букв, 1 пробел, 2 знака препинания)*

*i = 16 бит = 2 байта*

*I = k⋅i = 16⋅2 = 32 байта*

*Ответ: 32 байта*

**Пример 2.**Сообщение, состоящее из 120 символов, занимает 135 байт памяти. Какова максимально возможная мощность алфавита, использованного для записи сообщения?

*Решение:*

*k = 120 символов*

*I = 135 байт = 1080 бит*

*i =*I/k*​ =*1080/*120​ = 9 бит на 1 символ*

*N = 2i = 29 = 512 символов*

*Ответ: 512 символов*

**Пример 3.**Рассказ, набранный на компьютере, содержит 12 страниц, на каждой странице 48 строк, в каждой строке 64 символа. Определите информационный объём рассказа в килобайтах, если каждый символ кодируется 16 битами.

*Решение:*

*k = 12⋅48⋅64 = 36864 символа*

*i = 16 бит = 2 байта*

*I = k⋅i = 36864⋅2​/1024= 72 Кб*

*Ответ: 72 Кб*

**Пример 4. Информационный объём статьи составляет 32 Кбайт. Сколько страниц займет статья, если на одной странице помещается 32 строки по 64 символа, а каждый символ занимает 8 бит памяти?**

*Решение:*

*I = 32 Кб*

*I = k⋅i*

*k = x⋅32⋅64*

*32⋅1024⋅8 = x⋅32⋅64⋅8  
 i = 8 бит           x = 32⋅1024⋅8​/32⋅64⋅8 = 16*

*Ответ: 16 страниц*

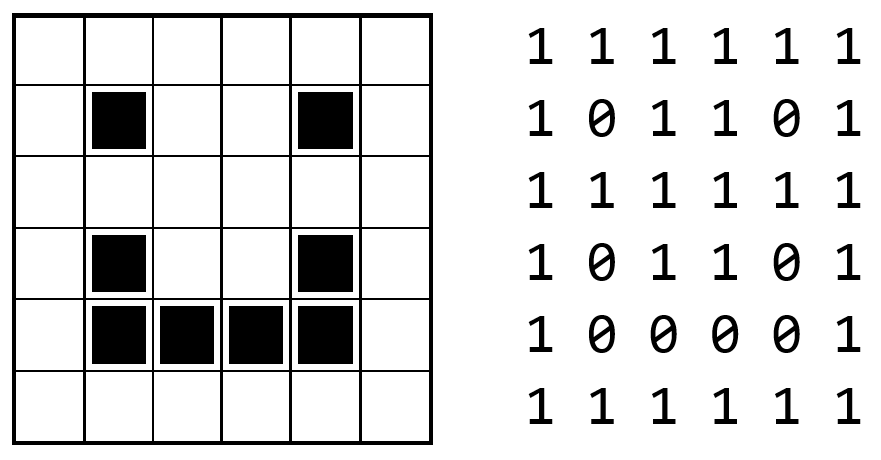
Таким образом, мы познакомились с кодировками ASCII и Unicode, научились измерять информационный объём текста.

# Измерение объёма графической информации

### Растровые изображения

Говоря об изображениях, мы должны вспомнить, как происходит их дискретизация, то есть разбиение на равные части. Изображения, хранимые и обрабатываемые компьютером, называются **растровыми**  и состоят из **пикселей**(англ. pixel - **pic**ture **el**ement) – квадратных элементов, имеющих определенный цвет. Именно в таком виде мы сможем легко закодировать и затем определить количество информации в изображении.

При хранении растрового изображения в памяти компьютера последовательно хранятся коды каждого пикселя. В простейшем случае, если мы кодируем чёрно-белое растровое изображение, то чёрный цвет кодируется нулём, а белый цвет – единицей. Каждый пиксель заменяется на двоичный разряд в зависимости от его цвета и получается закодированное изображение. В данном случае оно будет занимать 36 бит, так как количество пикселей по горизонтали и по вертикали равно 6, а 1 пиксель занимает 1 бит.



### Глубина цвета

Кроме чёрно-белых изображений мы можем хранить изображения и с более обширной палитрой. Тогда нам потребуется больше бит на 1 цвет. Вычислить это количество можно с помощью уже известной нам формулы Хартли:

2i = *N*

N – количество цветов в палитре  
i – глубина цвета (количество бит на 1 цвет)

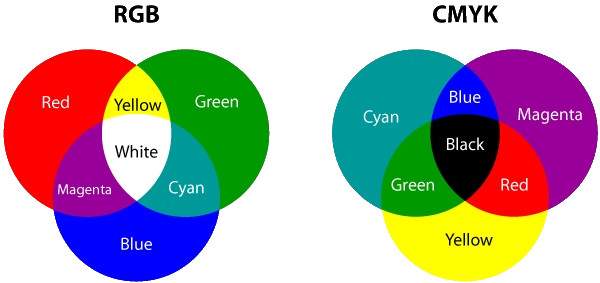
Наиболее распространёнными на данный момент являются палитры с глубиной цвета 8, 16, 24 и 32 бита.

|  |  |
| --- | --- |
| Глубина цвета | Количество цветов в палитре |
| 8 | 28 = 256 |
| 16 | 216 = 65536 |
| 24 | 224 = 16777216 |
| 32 | 232 = 4294967296 |

### 

### Цветовые палитры

Несмотря на то, что количество и набор цветов в палитре может быть произвольным, компьютер должен заранее знать, как именно кодируется тот или иной цвет. Тут нам на помощь приходит физика и то, как человеческий глаз воспринимает цвет. Ещё в 1861 году физик Джеймс Максвелл предложил использовать аддитивный синтез цвета, то есть получать любые оттенки путём смешения базовых цветов.

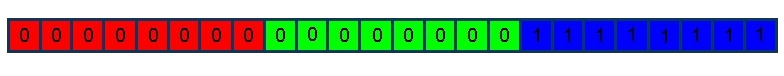


Базовыми цветами выбраны красный, зелёный и синий, отсюда и название цветовой модели RGB – по первым буквам названий цветов, red green blue. Изначально мы имеем черный цвет. Чёрный цвет – это ноль, ничего, отсутствие какого-либо цвета. Добавляя к нему базовые цвета в разных пропорциях будем получать нужный оттенок. Если смешать все три базовых цвета, то получим белый. Существует также обратная цветовая модель CMYK, которая образуется путём вычитания из белого. Базовыми цветами являются cyan (бирюзовый), magenta (пурпурный) и yellow (жёлтый). Если смешать все 3 цвета, то получим чёрный. Основное различие между этими цветовыми моделями в том, что RGB используется для представления цвета на экране, а CMYK – при печати изображений (т.к. печатаем мы обычно на белом, а не на чёрном).

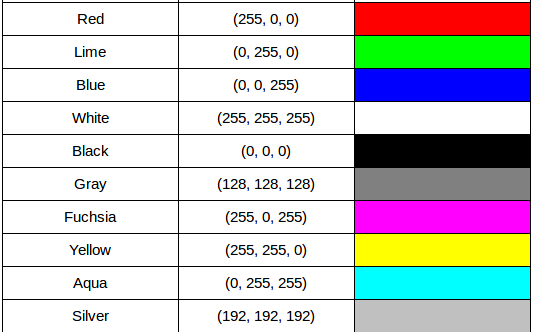
### 

### **Палитра RGB**

Для кодирования цвета в палитре RGB отводится 3 байта: по 1 байту на каждый из базовых цветов. Диапазон значений для каждого цвета может изменяться от 0 до 255 (максимальное десятичное число, которое поместится в 1 байт). Всего таким образом можно закодировать более 16 миллионов различных цветов.



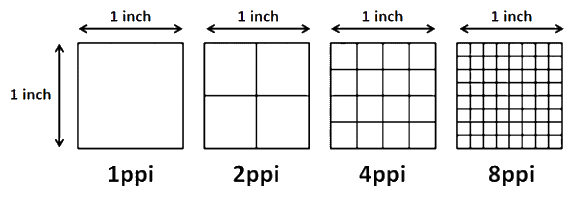
Вот некоторые примеры того, какие оттенки получатся, если смешивать базовые цвета в различных пропорциях. Отметим, что чем ближе значения к максимальным, тем более насыщенный цвет мы получим. Если все три значения равны 0, то получаем чёрный, а если все три максимальны и равны 255, то получаем белый цвет.



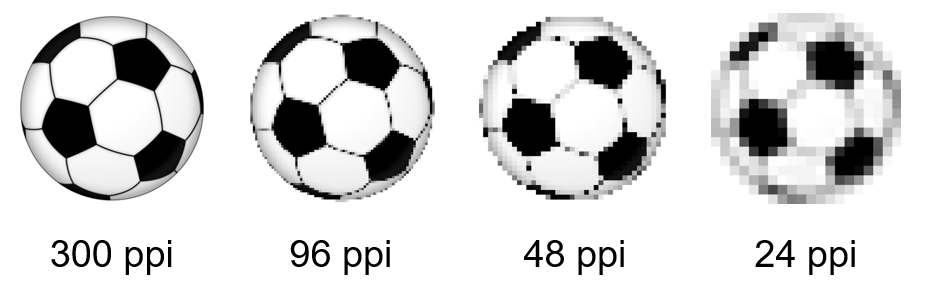
### 

### **Разрешение**

Кроме цветовой палитры, информационный объём изображения будет зависеть от разрешения, то есть количества пикселей по каждой из сторон. Часто разрешение обозначают плотностью расположения пикселей на 1 дюйм, такая характеристика **ppi**(pixels per inch).- пиксилей на дюйм



Например, если дано изображение с разрешением 4 ppi, то каждый квадратный дюйм изображения будет представлять собой квадрат 4 на 4 пикселя.

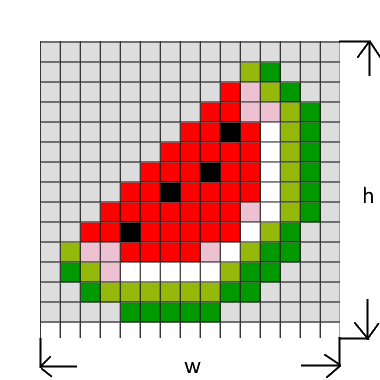


Очевидно, что чем больше разрешение, тем выше качество изображения. Чем больше пикселей, тем меньший физический размер они имеют. При разрешении в несколько сотен ppi пиксели уже не различимы.

### 

### Информационный объём изображения

Информационный объём изображения зависит от количества пикселей и глубины цвета и вычисляется по формуле:



**I =** *w***⋅***h***⋅***i*

w - количество пикселей по горизонтали  
          h - количество пикселей по вертикали  
          i - глубина цвета (бит)

Рассмотрим примеры задач на нахождение информационного объёма изображений.

***Пример 1.****Определите информационный объём изображения размером 1024х768 пикселей, палитра которого состоит из 64 цветов.*

*Решение:*

*N = 64*

*2i = 64*

*i = 6 бит*

*I = w⋅h⋅i = 1024⋅768⋅6 = 4718592 бит = 576 Кбайт*

*Ответ: 576 Кбайт*

***Пример 2.****Фотография размером 2048х800 пикселей занимает в памяти 600 Кбайт. Сколько цветов в палитре изображения?*

*Решение:*

I *=k⋅i*

*k - количество символов в тексте*

*i - информационный объём одного символа*

*I = w⋅h⋅i  
2048⋅800⋅i = 600⋅1024⋅8  
i = 600⋅1024⋅8 / 2048⋅800​ = 3 бита  
N = 23 = 8 цветов*

*Ответ: 8 цветов****Пример 3.****Найдите информационный объём изображения с разрешением 200 ppi размером 3х4 дюйма в палитре RGB, i = 3 байта*

*Решение:*

*i = 3 байта  
w = 3⋅200 = 600  
h = 4⋅200 = 800  
I = 600⋅800⋅3 = 1440000 байт = 1406,25 Кбайт = 1,37 Мбайт*

*Ответ: 1,37 Мбайт****Пример 4.****Файл с изображением занимает 480 Кбайт. Известно, что его разрешение уменьшили с 600 ppi до 150 ppi, а количество цветов в палитре увеличили с 16 до 256. Определите информационный объём измененного файла.*

*Решение:*

*w1⋅h1 = 600x600   
w2⋅h2 = 150x150 (в 16 раз меньше)  
N1 = 16 цветов → i1 = 4 бита  
N2 = 256 цветов → i2 = 8 бит (в 2 раза больше)  
I2 = I1​⋅2/16​ = 480⋅2/16​ = 60 Кбайт*

*Ответ: 60 Кбайт*

Мы рассмотрели основные принципы кодирования и хранения растровых изображений, узнали о цветовых схемах, разрешении, научились решать задачи на нахождение информационного объёма графических файлов.

**Вопросы для самоконтроля:**

1. Какие символьные кодировки вам известны? В чём их особенности и отличия?
2. Какие основные группы символов присутствуют в большинстве кодировок?
3. От чего зависит информационный объём текста?
4. Что общего между мозаичными изображениями и формированием изображения на экране монитора?
5. Какие особенности нашего зрения положены в основу формирования изображений на экране компьютера?
6. Опишите цветовую модель RGB.