Theory1-信息的编码、存储与管理

数制及二进制运算

如果两个有理数相等，则这两个数的整数部分和小数部分一定分别相等

熟练掌握　十进制⇔N 进制

整数部分　除 N 取余

2 | 253 ……1 低位
2 | 126 ……0
2 | 63 ……1
2 | 31 ……1
2 | 15 ……1
2 | 7 ……1
2 | 3 ……1
2 | 1 ……1 高位
0

小数部分　乘 N 取整

0.745×2 = 1.490
0.490×2 = 0.980
0.980×2 = 1.960
0.960×2 = 1.920
0.920×2 = 1.840
0.840×2 = 1.680

$(253)_{10} = (11111101)_2$　　$(0.745)_{10} \approx (0.101111)_2$

十进制小数一般不能精确表示成有限位的二进制小数

基本方法

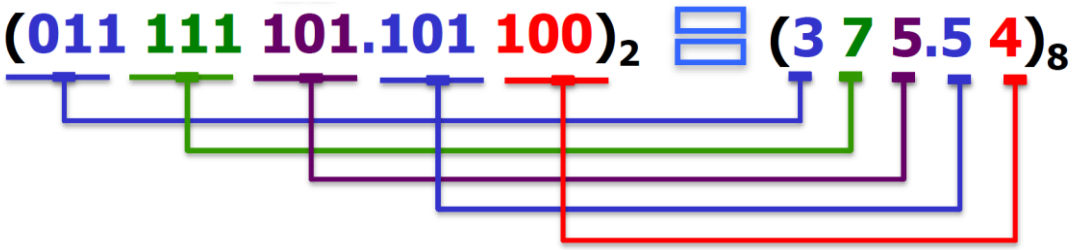
把非十进制数各个数位上的数按权值展开求和

7 6 5 4 3 2 1 0 -1 -2 -3 -4

$(1\ 1\ 1\ 1\ 1\ 1\ 0\ 1.1\ 0\ 1\ 1)_2$

$1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+0\times2^{-3}+1\times2^{-4}$

其他的转换一般通过十进制作为中介
二进制⇔八进制　以及　二进制⇔十六进制 则不需要



二进制的算数运算

二进制加法运算规则

- 0+0=0
- 0+1=1+0=1
- 1+1=10（进位）

$$\begin{array}{r} (1101)_2 \\ +(1011)_2 \\ \hline (11000)_2 \end{array}$$

乘法示例

$$\begin{array}{r} (1101)_2 \\ \times (1011)_2 \\ \hline 1101 \\ 1101 \\ 0000 \\ 1101 \\ \hline (10001111)_2 \end{array}$$

验算：转换成十进制后计算，之后再转换为二进制。

二进制的逻辑运算

- 逻辑非：~
  - ✓~1 = 0; ~0 = 1
- 逻辑或：|
  - ✓0 | 0 = 0; 0 | 1 = 1; 1 | 0 = 1; 1 | 1 = 1
- 逻辑与：&
  - ✓0 & 0 = 0; 0 & 1 = 0; 1 & 0 = 0; 1 & 1 = 1
- 逻辑异或：^
  - ✓0 ^ 0 = 0; 0 ^ 1 = 1; 1 ^ 0 = 1; 1 ^ 1 = 0

按位运算，没有进位

```
~11010    |01101    &01101    ^01101
 00101     11101     01001     10100
```

其他信息编码

一切信息进入计算机，都转换为二进制串

编号	中文名称	英文名称/缩写	物理含义
1	位/比特	bit	一个基本存储单元，只能存放一个0或一个1
2	字节	Byte	8个bit
3	千字节	KB	2^{10}个Byte
4	兆字节	MB	2^{20}个Byte，即2^{10}个KB
5	千兆字节	GB	2^{30}个Byte，即2^{10}个MB
6	兆兆字节	TB	2^{40}个Byte，即2^{10}个GB

整数采用原码、反码和补码以及它们的表示范围

补码带符号整数的表示范围：32 位整数为-2^{31}~ $2^{31}-1$，64 位整数表示为-2^{63}~ $2^{63}-1$

浮点数大多采用 IEEE 754 标准：单精度 32 位表示，8 位指数 23 位尾数；范围约 (10^{-45} ~ 10^{38})，约 7 位精度 10 进制有效数字

英文字符编码大多采用 ASCII 码，每个字符用 7 位二进制数($d_6d_5d_4d_3d_2d_1d_0$)来表示；可表示 128 个字符(7 位二进制共有 128 种状态，2^7= 128）————Python 内置函数 ord, chr

汉字编码用两个字节编码一个汉字

声音编码离散采样

颜色编码

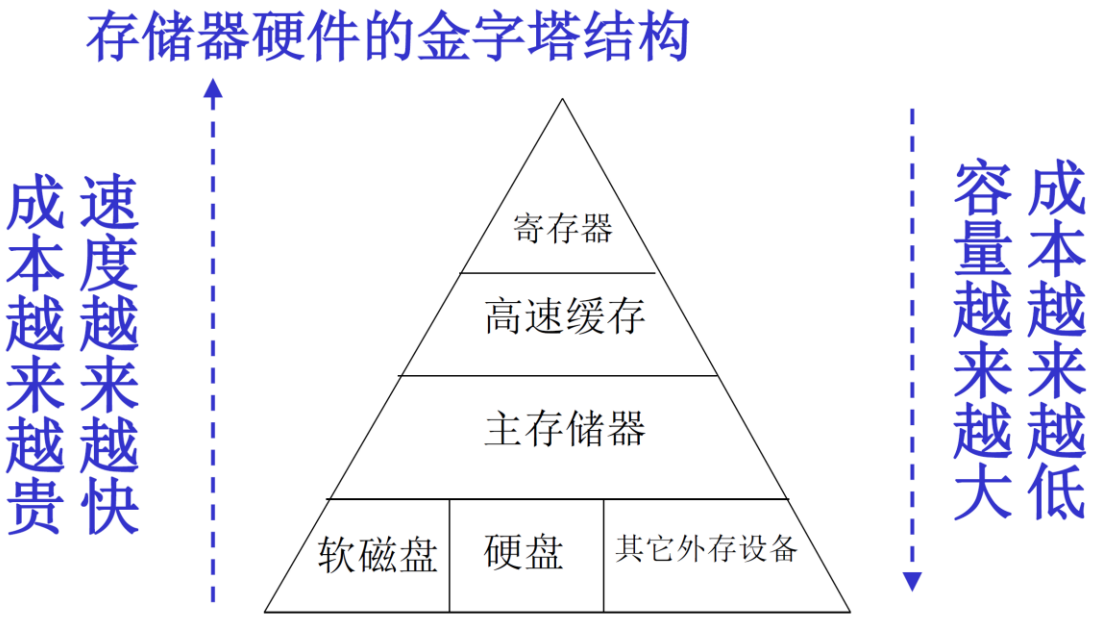
颜色系统：
- 单色系统：2色 (1位)
- 灰度系统
- 彩色系统
  - 16色 (4位)
  - 256色 (8位)
  - 真彩色 (16位 / 24位)

图像编码：1.点阵表示　　2.矢量表示

影像编码：连续图像存储

存储 10 分钟的 640×480 的真彩色(24 位)连续影像，按照每秒钟 25 帧计算，不包括声音信息，需要（640×480×3 byte×25 帧×10 分钟×60 秒）字节，大约 13GB(13184M) 字节。

分层存储



为什么要分为内外存（存储墙问题）？
由于 CPU 处理速度非常快，而向存储器中读写数据的速度却相对较慢，造成对 CPU 处理能力的浪费。

局部性原理：

时间局部性（Temporal Locality）：如果一个信息项正在被访问，那么在近期它很可能还会被再次访问。程序循环、堆栈等是产生时间局部性的原因。

空间局部性（Spatial Locality）：在最近的将来将用到的信息很可能与现在正在使用的信息在空间地址上是临近的。

用局部性原理解决存储墙问题

信息的输入和输出

信息存储的原理和设备（存储介质、存储器）

信息的管理（文件系统、数据库）

（太杂，自行看一遍 PPT~）