编码分析

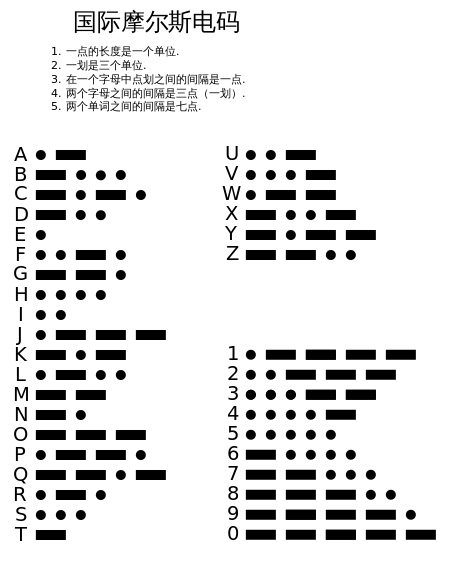
# 通信领域常用编码

## 1、电话拨号编码

1-9 分别使用 1-9 个脉冲，0 则表示使用 10 个脉冲。

## 2、Morse 编码

参见 [摩尔斯编码 - 维基百科](https://zh.wikipedia.org/wiki/%E6%91%A9%E5%B0%94%E6%96%AF%E7%94%B5%E7%A0%81)，对应表如下



### 特点 :

* 只有 . 和 -；
* 最多 6 位；
* 也可以使用 01 串表示。

### 工具 :

* [摩尔斯编码在线转换](http://www.zhongguosou.com/zonghe/moErSiCodeConverter.aspx)

### 题目 :

* JarvisOJ - Basec - 「-.- 字符串」

## 3、敲击码

敲击码（Tap code）是一种以非常简单的方式对文本信息进行编码的方法。因该编码对信息通过使用一系列的点击声音来编码而命名，敲击码是基于 5 ×5 方格波利比奥斯方阵来实现的，不同点是是用 K 字母被整合到 C 中。

# Tap Code 1 2 3 4 5

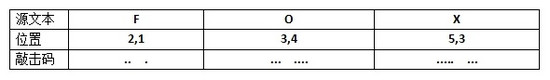
# 1 A B C/K D E

# 2 F G H I J

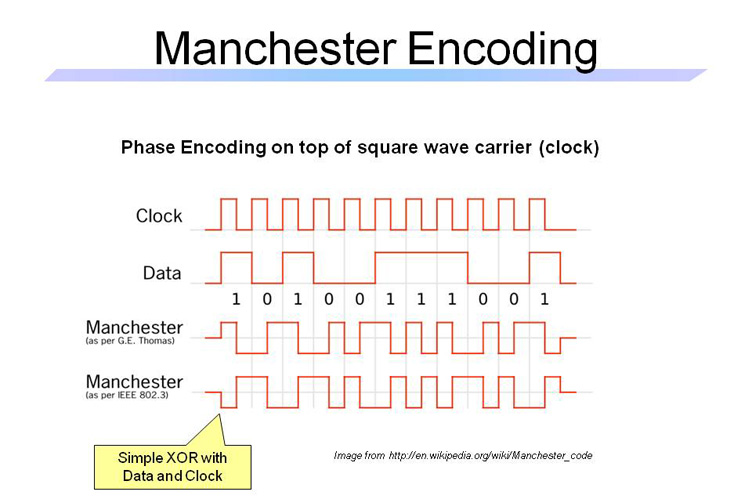
# 3 L M N O P

# 4 Q R S T U

# 5 V W X Y Z



## 4、曼彻斯特编码

* [曼彻斯特编码 - 维基百科](https://zh.wikipedia.org/wiki/%E6%9B%BC%E5%BD%BB%E6%96%AF%E7%89%B9%E7%BC%96%E7%A0%81)
* 在[电信](https://zh.wikipedia.org/wiki/%E7%94%B5%E4%BF%A1" \o "信号)与[数据存储](https://zh.wikipedia.org/wiki/%E9%9B%BB%E8%85%A6%E6%95%B8%E6%93%9A%E5%AD%98%E8%B2%AF%E5%99%A8" \o ")中, **曼彻斯特编码**（**Manchester coding**），又称**自同步码**、**相位编码**（**phase encoding**，**PE**），能够用[信号](https://zh.wikipedia.org/wiki/%E4%BF%A1%E5%8F%B7" \o ")的变化来保持发送设备和接收设备之间的[同步](https://zh.wikipedia.org/wiki/%E5%90%8C%E6%AD%A5" \o "同步)。它用[电压](https://zh.wikipedia.org/wiki/%E9%9B%BB%E5%A3%93" \o "电压)的变化来分辨0和1，从高电平到低电平的跳变代表1，而从低电平到高电平的跳变代表0(as per G.E.Tomas编码方式)。从高电平到低电平的跳变代表0，而从低电平到高电平的跳变代表1(as per IEEE 802.3编码方式)，下方有所展示。信号的保持不会超过一个[比特位](https://zh.wikipedia.org/wiki/%E6%AF%94%E7%89%B9" \o "比特)的时间间隔。即使是0或1的序列，信号也将在每个时间间隔的中间发生跳变。这种跳变将允许接收设备的时钟与发送设备的时钟保持一致。
* 曼彻斯特编码（Manchester Encoding），也叫做相位编码(PE)，是一个同步时钟编码技术，被物理层使用来编码一个同步位流的时钟和数据。曼彻斯特编码被用在以太网媒介系统中。曼彻斯特编码提供一个简单的方式给编码简单的二进制序列而没有长的周期没有转换级别，因而防止时钟同步的丢失，或来自低频率位移在贫乏补偿的模拟链接位错误。在这个技术下，实际上的二进制数据被传输通过这个电缆，不是作为一个序列的逻辑1或0来发送的（技术上叫做反向不归零制(NRZ)）。
* 相反地，这些位被转换为一个稍微不同的格式，它通过使用直接的二进制编码有很多的优点。曼彻斯特编码，常用于局域网传输。在曼彻斯特编码中，每一位的中间有一跳变，位中间的跳变既作 时钟信号 ，又作数据信号；从低到高跳变表示"0"，从高到低跳变表示"1"。还有一种是差分曼彻斯特编码 ，每位中间的跳变仅提供时钟定时，而用每位开始时有无跳变表示"0"或"1"，有跳变为"0"，无跳变为"1"。<https://blog.csdn.net/Sun19910114/article/details/40016383>
* 
* **一.曼彻斯特编码：**
* 0：低——高 1：高——低
* **二.差分曼彻斯特编码：**
* 0表示有跳变 1表示无跳变
* 100110用曼彻斯特编码怎么表示

如果定义1->10,0->01则曼彻斯特码为：100101101001如果定义0->10,1->01则曼彻斯特码为：011010010110

## 5、格雷编码

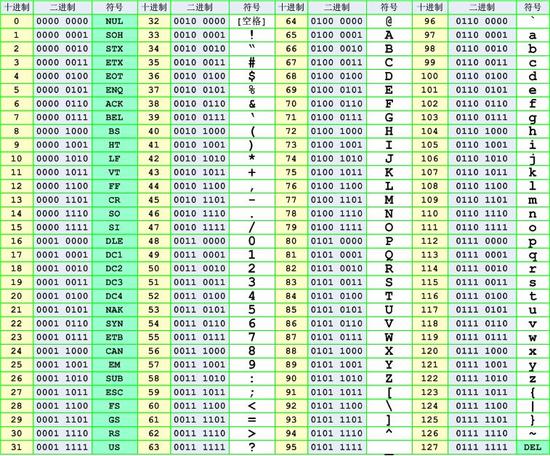
* [格雷码 - 维基百科](https://zh.wikipedia.org/wiki/%E6%A0%BC%E9%9B%B7%E7%A0%81)
* ****特征特点****：由相邻两位数之间只有一个位元改变的二进制数码构成数列集合。传统的二进位系统例如数字3的表示法为011，要切换为邻近的数字4，也就是100时，装置中的三个位元都得要转换，因此于未完全转换的过程时装置会经历短暂的，010,001,101,110,111等其中数种状态，也就是代表着2、1、5、6、7，因此此种数字编码方法于邻近数字转换时有比较大的误差可能范围。格雷码的发明即是用来将误差之可能性缩减至最小。
* 
* ****在线解密&工具****： <http://www.ab126.com/system/2780.html>
* [浅谈格雷编码](https://blog.csdn.net/xiaok0707/article/details/109013051)

# 计算机相关的编码

## 1、字母表编码

* A-Z/a-z 对应 1-26 或者 0-25

## 2、ASCII 编码



### 特点

我们一般使用的 ascii 编码的时候采用的都是可见字符，而且主要是如下字符

* 0-9, 48-57
* A-Z, 65-90
* a-z, 97-122

### 变形

#### **二进制编码 ：**

将 ascii 码对应的数字换成二进制表示形式。

* 只有 0 和 1
* 不大于 8 位，一般 7 位也可以，因为可见字符到 127。
* 其实是另一种 ascii 编码。

#### **十六进制编码 ：**

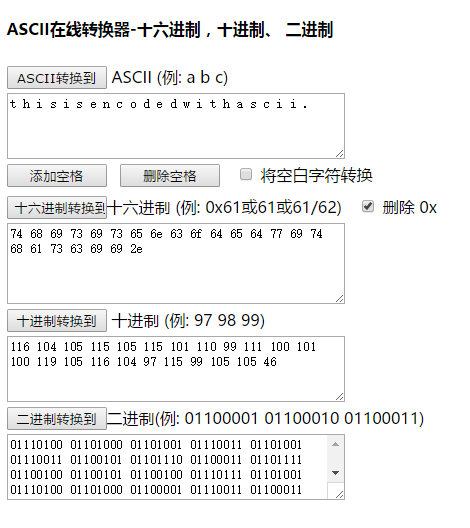
将 ascii 码对应的数字换成十六进制表示形式。

* A-Z→41-5 A
* a-z→61-7 A

### 工具

* jpk, ascii to number, number to ascii
* <http://www.ab126.com/goju/1711.html>

### 例子



### 2018 DEFCON Quals ghettohackers: Throwback

题目描述如下

Anyo!e!howouldsacrificepo!icyforexecu!!onspeedthink!securityisacomm!ditytop!urintoasy!tem!

第一直觉应该是我们去补全这些叹号对应的内容，从而得到 flag，但是补全后并不行，那么我们可以把源字符串按照 ! 分割，然后字符串长度 1 对应字母 a，长度 2 对应字母 b，以此类推

ori ='Anyo!e!howouldsacrificepo!icyforexecu!!onspeedthink!securityisacomm!ditytop!urintoasy!tem!'

sp = ori.split('!')

print repr(''.join(chr(97 + len(s) - 1) for s in sp))

进而可以得到，这里同时需要假设 0 个字符为空格。因为这正好使得原文可读。

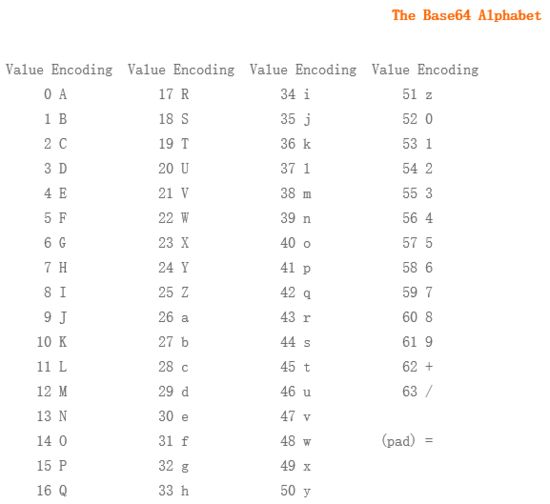
dark logic

### 题目

* Jarvis-basic - 德军的密码

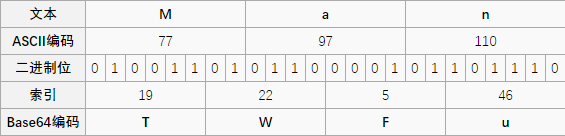
## 3、Base 编码

base xx 中的 xx 表示的是采用多少个字符进行编码，比如说 base64 就是采用以下 64 个字符编码，由于 2 的 6 次方等于 64，所以每 6 个比特为一个单元，对应某个可打印字符。3 个字节就有 24 个比特，对应于 4 个 Base64 单元，即 3 个字节需要用 4 个可打印字符来表示。它可用来作为电子邮件的传输编码。在 Base64 中的可打印字符包括字母 A-Z、a-z、数字 0-9，这样共有 62 个字符，此外两个可打印符号在不同的系统中而不同。

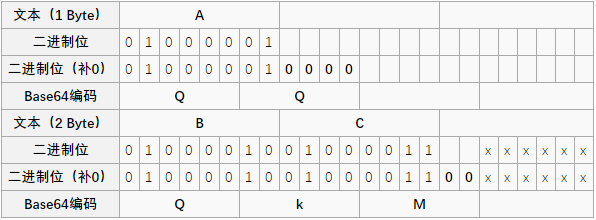


具体介绍参见 [Base64 - 维基百科](https://zh.wikipedia.org/wiki/Base64)

**编码 man**

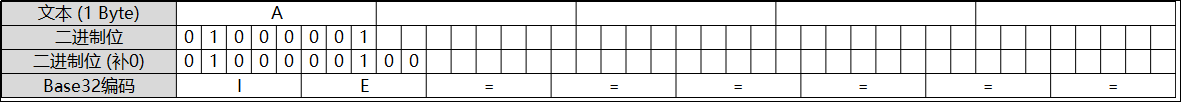
****

如果要编码的字节数不能被 3 整除，最后会多出 1 个或 2 个字节，那么可以使用下面的方法进行处理：先使用 0 值在末尾补足，使其能够被 3 整除，然后再进行 base64 的编码。在编码后的 base64 文本后加上一个或两个 = 号，代表补足的字节数。也就是说，当最后剩余一个八位字节（一个 byte）时，最后一个 6 位的 base64 字节块有四位是 0 值，最后附加上两个等号；如果最后剩余两个八位字节（2 个 byte）时，最后一个 6 位的 base 字节块有两位是 0 值，最后附加一个等号。参考下表：



由于解码时补位的 0 并不参与运算，可以在该处隐藏信息。

与 base64 类似，base32 使用 32 个可见字符进行编码，2 的 5 次方为 32，所以每 5 bit 为 1 个分组。5 字节为 40 bit，对应于 8 个 base32 分组，即 5 个字节用 8 个 base32 中字符来表示。但如果不足 5 个字节，则会先对第一个不足 5 bit 的分组用 0 补足了 5 bit ，对后面剩余分组全部使用 “=” 填充，直到补满 5 个字节。由此可知，base32 最多只有 6 个等号出现。例如：



### 特点

* base64 结尾可能会有 = 号，但最多有 2 个
* base32 结尾可能会有 = 号，但最多有 6 个
* 根据 base 的不同，字符集会有所限制
* **有可能需要自己加等号**
* **= 也就是 3D**
* 更多内容请参见 [base rfc](https://tools.ietf.org/html/rfc4648)

### 工具

* <http://www1.tc711.com/tool/BASE64.htm>
* python 库函数
* [读取隐写信息脚本](https://github.com/cjcslhp/wheels/tree/master/b64stego)

### 例子

题目描述参见 ctf-challenge中 [misc 分类的 base64-stego 目录](https://github.com/ctf-wiki/ctf-challenges/tree/master/misc/encode/computer/base64-stego)中的 data.txt 文件。

使用脚本读取隐写信息。

import base64

def deStego(stegoFile):

b64table = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

with open(stegoFile,'r') as stegoText:

message = ""

for line in stegoText:

try:

text = line[line.index("=") - 1:-1]

message += "".join([ bin( 0 if i == '=' else b64table.find(i))[2:].zfill(6) for i in text])[2 if text.count('=') ==2 else 4:6]

except:

pass

return "".join([chr(int(message[i:i+8],2)) for i in range(0,len(message),8)])

print(deStego("text.txt"))

输出:flag{BASE64\_i5\_amaz1ng}

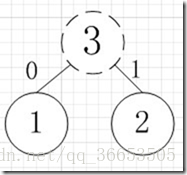
## 4、霍夫曼编码

参见 [霍夫曼编码](https://zh.wikipedia.org/wiki/%E9%9C%8D%E5%A4%AB%E6%9B%BC%E7%BC%96%E7%A0%81)。

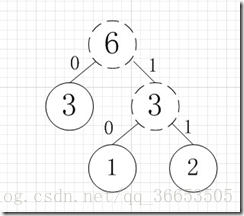
哈夫曼编码(Huffman Coding)，又称霍夫曼编码，是一种编码方式，可变字长编码(VLC)的一种。Huffman于1952年提出一种编码方法，该方法完全依据字符出现概率来构造异字头的平均长度最短的码字，有时称之为最佳编码，一般就叫做Huffman编码（有时也称为霍夫曼编码）。

哈夫曼编码，主要目的是根据使用频率来最大化节省字符（编码）的存储空间。

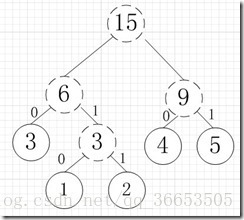
简易的理解就是，假如我有A,B,C,D,E五个字符，出现的频率（即权值）分别为5,4,3,2,1,那么我们第一步先取两个最小权值作为左右子树构造一个新树，即取1，2构成新树，其结点为1+2=3，如图：



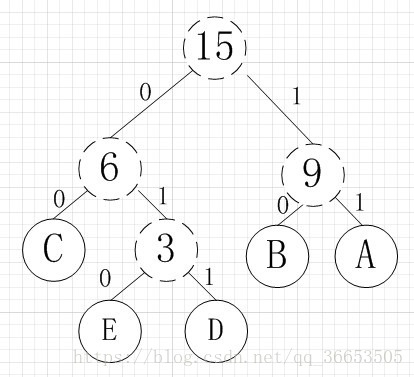
虚线为新生成的结点，第二步再把新生成的权值为3的结点放到剩下的集合中，所以集合变成{5,4,3,3}，再根据第二步，取最小的两个权值构成新树，如图：



再依次建立哈夫曼树，如下图：



其中各个权值替换对应的字符即为下图：



所以各字符对应的编码为：A->11,B->10,C->00,D->011,E->010

霍夫曼编码是一种无前缀编码。解码时不会混淆。其主要应用在数据压缩，加密解密等场合。

如果考虑到进一步节省存储空间，就应该将出现概率大（占比多）的字符用尽量少的0-1进行编码，也就是更靠近根（节点少），这也就是最优二叉树-哈夫曼树。

原文链接：<https://blog.csdn.net/qq_36653505/article/details/81701181>

**题目**

**[哈夫曼树](https://blog.csdn.net/qq_43625917/article/details/96152962)**

## 5、XXencoding

XXencode 将输入文本以每三个字节为单位进行编码。如果最后剩下的资料少于三个字节，不够的部份用零补齐。这三个字节共有 24 个 Bit，以 6bit 为单位分为 4 个组，每个组以十进制来表示所出现的数值只会落在 0 到 63 之间。以所对应值的位置字符代替。



具体信息参见[维基百科](https://en.wikipedia.org/wiki/Xxencoding)

### 特点

* 只有数字，大小写字母
* + 号，- 号。

### **工具**

* <http://web.chacuo.net/charsetxxencode>

## 6、URL 编码

参见[URL 编码 - 维基百科](https://zh.wikipedia.org/wiki/%E7%99%BE%E5%88%86%E5%8F%B7%E7%BC%96%E7%A0%81)。

****特征特点****：一个字符ascii码的十六进制，然后在前面加上%

****在线解密&工具****： <http://ctf.ssleye.com/url.html>

## 7、Unicode 编码

参见[Unicode - 维基百科](https://zh.wikipedia.org/wiki/Unicode)。

****特征特点****：国际标准字符，将全球的各种语言的每个字符定义一个唯一的编码，以满足跨语言、跨平台文本信息的转换，当编码和解码的字符集出现不一致的时候就会出现乱码。

注意，它有四种表现形式。

**例子**

源文本： The

&#x [Hex]: &#x0054;&#x0068;&#x0065;

&# [Decimal]: &#00084;&#00104;&#00101;

\U [Hex]: \U0054\U0068\U0065

\U+ [Hex]: \U+0054\U+0068\U+0065

**工具**

<http://tool.chinaz.com/tools/unicode.aspx>

## HTML 实体编码

**HTML 中的预留字符必须被替换为字符实体。**

## **HTML 实体**

在 HTML 中，某些字符是预留的。

在 HTML 中不能使用小于号（<）和大于号（>），这是因为浏览器会误认为它们是标签。

如果希望正确地显示预留字符，我们必须在 HTML 源代码中使用字符实体（character entities）。

字符实体类似这样：

&*entity\_name*;

或者

&#*entity\_number*;

如需显示小于号，我们必须这样写：&lt; 或 &#60;

**提示：**使用实体名而不是数字的好处是，名称易于记忆。不过坏处是，浏览器也许并不支持所有实体名称（对实体数字的支持却很好）。

## **不间断空格（non-breaking space）**

HTML 中的常用字符实体是不间断空格(&nbsp;)。

浏览器总是会截短 HTML 页面中的空格。如果您在文本中写 10 个空格，在显示该页面之前，浏览器会删除它们中的 9 个。如需在页面中增加空格的数量，您需要使用 &nbsp; 字符实体。

## **HTML 实例示例**

用 HTML 实体符号做实验：[亲自试一试](https://www.w3school.com.cn/tiy/t.asp?f=eg_html_entities" \t "https://www.w3school.com.cn/html/_blank)

## **HTML 中有用的字符实体**

**注释：**实体名称对大小写敏感！



如需完整的实体符号参考，请访问 [HTML 实体符号参考手册](https://www.w3school.com.cn/tags/html_ref_entities.html" \o "HTML ISO-8859-1 参考手册)。

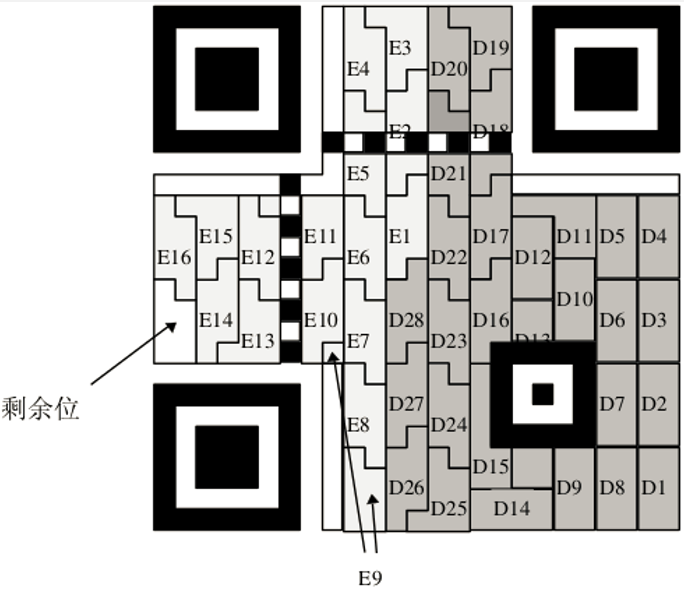
**三、现实世界中常用的编码**

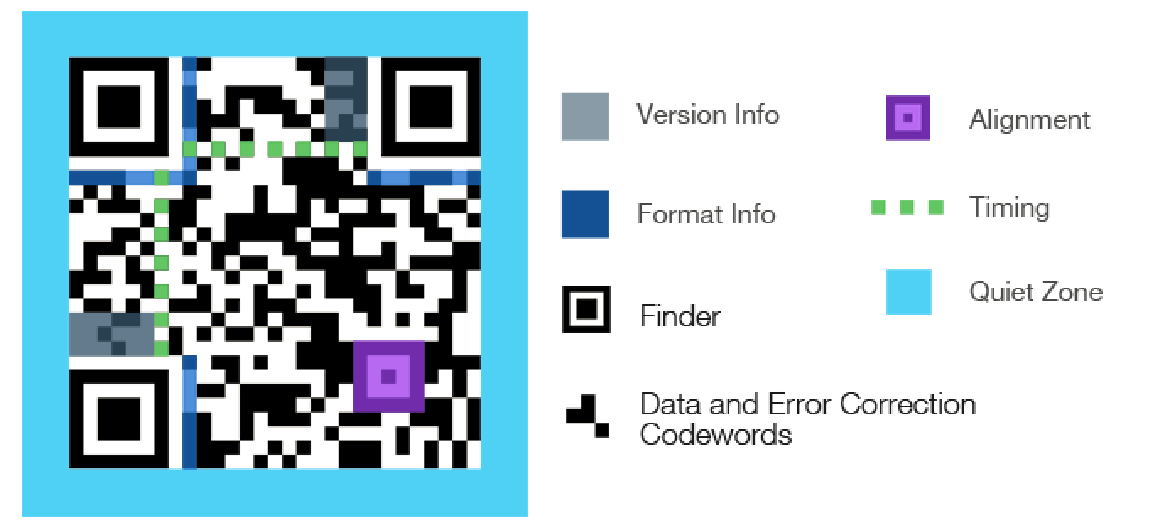
## 1、条形码

* 宽度不等的多个黑条和空白，按照一定的编码规则排列，用以表达一组信息的图形标识符
* 国际标准
* EAN-13 商品标准，13 位数字
* Code-39：39 字符
* Code-128：128 字符
* [条形码在线识别](https://online-barcode-reader.inliteresearch.com)

## 2、二维码

* 用某种特定几何图形按一定规律在平面分步的黑白相间的图形记录数据符号信息
* 堆叠式 / 行排式二维码：Code 16 k、Code 49、PDF417
* 矩阵式二维码：QR CODE





# 汉信码

[汉信码维基百科](https://zh.wikipedia.org/zh-cn/%E6%B1%89%E4%BF%A1%E7%A0%81)

**汉信码**是一种矩阵式[二维条码](https://zh.wikipedia.org/wiki/%E4%BA%8C%E7%BB%B4%E6%9D%A1%E7%A0%81" \o "二维条码)。从形状上，它呈正方向、有深色和浅色数据模块分布其间。

# 工具：

010editor

Stegsolve.jar

Excel

在线汉信码识别网站：<http://www.efittech.com/hxdec.html>

# 例题：2020网鼎杯-青龙组-虚幻2赛题

# ChaMd5安全团队bilibili视频讲解:<https://www.bilibili.com/video/av883175933/>

1619455-20200516120720521-645268486

### ****其他：****

****● 当铺密码****

****特征特点****：密文由[0-9]数字组成（以当前汉字有多少笔画出头，就是转化成对应的数字） 密文：0123456789 原文：口由中人工大王夫井羊 ****在线解密&工具****： <http://www.zjslove.com/3.decode/dangpu/index.html>

****● 核心价值观编码****

****特征特点****：密文特征为社会主义核心价值观的关键字 ****在线解密&工具****： <http://ctf.ssleye.com/cvencode.html>

****● 与佛论禅****

****特征特点****：密文以”佛曰：如是我闻：”开头，密文一般是与关佛经的汉字 ****在线解密&工具****： <http://www.keyfc.net/bbs/tools/tudoucode.aspx>

**网上其他整理**

<https://blog.csdn.net/m0_46230316/article/details/105356633>