2-2 基数と基数変換

我们日常使用的**十进制**，是使用 0～9 这 10 个数字，每一位都附加了 10 的幂次作为权重的数值。

(362.9)₁₀ ＝ 3×10² ＋ 6×10¹ ＋ 2×10⁰ ＋ 9×10⁻¹

＝ (362.9)₁₀

十进制的位权（10）称为**基数**，用基数 n 表示的数值称为**n 进制**。  
n 进制是指使用 0～(n−1) 这 n 个数字，当数值达到 n 时就向高位进 1 的表示方法。

1. 2進数

**二进制**是使用 0 和 1 这两个数字表示的数值，当达到 2 时向高位进 1。

在计算机内部，信息的最小单位是比特（bit），由于只能使用 0 和 1 这两个数字来表示信息，因此计算机使用二进制来表示数值。

(101.1)₂ ＝ 1×2² ＋ 0×2¹ ＋ 1×2⁰ ＋ 1×2⁻¹

＝ (5.5)₁₀

1. 8進数

**八进制**是使用 0～7 这 8 个数字表示的数值，当达到 8 时向高位进 1。

(317.5)₈ ＝ 3×8² ＋ 1×8¹ ＋ 7×8⁰ ＋ 5×8⁻¹

＝ (207.625)₁₀　※( )₈ 中的 8 表示该数是八进制数

1. 16進数

**十六进制**使用数字 0～9 以及 A（10）、B（11）、C（12）、D（13）、E（14）、F（15）这 16 个符号来表示，当数值达到 16 时进位。

(1A6.E)₁₆ ＝ 1×16² ＋ 10×16¹ ＋ 6×16⁰ ＋ 14×16⁻¹

＝ (422.875)₁₀

1. 2進数と8進数、16進数の関係

二进制与十进制相比，位数会非常多。在计算机内部处理时没有问题，但在人类阅读计算机内部状态时就很不方便。因此，为了让表示更容易理解，会使用八进制和十六进制。

八进制的每一位的权值是 8（= 2³） 的幂次

因此**二进制每 3 位信息对应八进制的 1 位。**

二进制：(1011100.11101)₂ → （001 011 100 . 111 010）₂

以小数点为基准，每 3 位分组（不足 3 位的部分用 0 补齐）  
→ 八进制：(1 3 4 . 7 2)₈

十六进制的每一位的权值是 16（= 2⁴） 的幂次

因此**二进制每 4 位信息对应十六进制的 1 位。**

二进制：(1011100.11101)₂ → （0101 1100 . 1110 1000）₂

以小数点为基准，每 4 位分组（不足 4 位的部分用 0 补齐）  
→ 十六进制：(5 C . E 8)₁₆

(5) 基数変換

像二进制、八进制、十六进制这样，基数之间存在固定的幂次关系时，如前文所述, 可以通过将多位分组的方式进行进位制转换。  
但是，如果像二八十六与十进制这样，基数之间不存在固定的幂次关系，就需要通过计算来进行转换。

**n 进制 → 十进制**

箱线图

AI 生成的内容可能不正确。每一位的数字 × n 的位权（n 的幂次），然后全部加起来

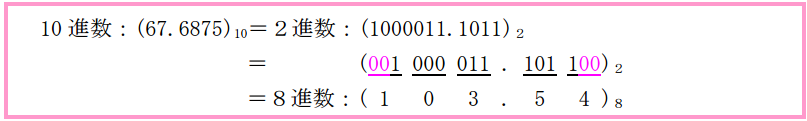
**十进制 → n 进制**

* **整数部分：不断除以 n，记录余数，倒序排列**
* **图示

  AI 生成的内容可能不正确。小数部分：不断乘以 n，记录整数部分，顺序排列**

**图片包含 示意图

AI 生成的内容可能不正确。**

当目标是 **8 进制或 16 进制** 时，可以先转成二进制，再按 3 位或 4 位一组去转换，这样会更快更直观.

**【无法用有限位数表示的小数】**

当将 10 进制的小数部分转换为 n 进制时，如果运算结果的小数部分始终不为 0 而不断循环，就可能出现 **无法转换成有限小数** 的情况。

墙上的钟表

AI 生成的内容可能不正确。例如，将 10 进制的 **(0.2)** 转换为 2 进制时，会如下所示：

由此可见，如果基数转换的结果是 **无限小数** 或 **循环小数**，计算机内部会将该数值作为 **近似值** 来处理。