187. 重复的DNA序列

题目描述:

所有 DNA 都由一系列缩写为 `'A'`，`'C'`，`'G'` 和 `'T'` 的核苷酸组成，例如：`"ACGAATTCCG"`。在研究 DNA 时，识别 DNA 中的重复序列有时会对研究非常有帮助。

编写一个函数来找出所有目标子串，目标子串的长度为 10，且在 DNA 字符串 `s` 中出现次数超过一次。

示例 1：

```
输入：s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"
输出：["AAAAACCCCC","CCCCCAAAAA"]
```

示例 2：

```
输入：s = "AAAAAAAAAAAAA"
输出：["AAAAAAAAAA"]
```

【二进制编码-减少空间消耗】【位运算-减少时间消耗】

添加末尾 1 很简单，和上面的思路一样：

- 左移以释放最后两位：`bitmask <<= 2`。
- 添加 1 到最后两位：`bitmask |= 1`。

现在的问题是删除前导 2。换句话说，问题是将 2L 位和 (2L + 1) 位设置为零。

我们可以使用一个技巧去重置第 n 位的值：`bitmask &= ~(1 << n)`。

这个技巧很简单：

- `1 << n` 是设置第 n 位为 1。
- `~(1 << n)` 是设置第 n 位为 0，且全部低位为 1。
- `bitmask &= ~(1 << n)` 是将 bitmask 第 n 位设置为 0。

技巧的简单使用方法是先设置第 2L 位，然后再设置 (2L + 1) 位：`bitmask &= ~(1 << 2 * L) & ~(1 << (2 * L + 1)`。可以简化为 `bitmask &= ~(3 << 2 * L)`：