作业提交服务器：ftp://192.168.126.193

用户名：stu

密码：stu1234

本次作业提交截止时间：10月31日23:59

❗注意事项

- 提交作业的算法

```
if 有多个文件：
  打包成zip文件
else: #只有一个pdf
    pass
用自己的学号将文件命名
上传至对应的文件夹中
```

- 代码请上传源文件，比如.py文件或.c文件，不要上传工程文件。
- 非代码请上传pdf，非pdf（比如doc文件、md文件、tex文件）请转换成pdf。
- 涉及公式的作业，推荐使用markdown编辑器，比如Typora，提交导出的pdf文档即可。
- 涉及算法伪代码的作业，推荐使用在线$LaTeX$编辑器Overleaf完成，提交编译生成的pdf文档即可。如果需要在overleaf中输入中文并能正确编译，首先在文档中加入package:`\usepackage[UTF8]{ctex}`，然后修改设置：设置->修改Latex引擎->选择"XeLatex"。

---

1. 1854 年，考古学家发现了公元前 2000 年左右雕刻的苏美尔泥板，上面列出了 59 以内整数的平方。这一发现使一些学者推测古代苏美尔人通过平方来进行乘法运算，比如使用形如 $x \cdot y = (x^2 + y^2 - (x-y)^2)/2$这样的等式。不幸的是，这些学者对苏美尔人如何对更大的数字进行平方保持沉默。四千年后，我们终于可以通过递归的力量将这些苏美尔数学家从苦役中解救出来！

   (a) 设计 Karatsuba 算法的一种变体，能够在 $O(n^{\log_2 3})$ 时间内计算任何n位数的平方，方法是将原问题归约到计算3个$\lceil n/2 \rceil$ 位数的平方。

   (b) 设计一个递归算法，能够在$O(n^{\log_3 6})$ 时间内计算任何n位数的平方，方法是将原问题归约到计算6个$\lceil n/3 \rceil$ 位数的平方。

   (c) 设计一个递归算法，能够在$O(n^{\log_3 5})$ 时间内计算任何n位数的平方，方法是将原问题归约到仅仅计算5个 $(n/3 + O(1))$ 位数的平方。提示：考虑$(a+b+c)^2 + (a-b+c)^2$。

2. 两个正整数 a 和 b的最大公约数$\gcd(a,b)$是使得 a/d和b/d 都是整数的最大整数 d。

   (a) 证明下列等式成立。

$$\gcd(a,b) = \begin{cases} 2\gcd(a/2, b/2) & \text{if } a,b \text{ are even} \\ \gcd(a, b/2) & \text{if } a \text{ is odd, } b \text{ is even} \\ \gcd((a-b)/2, b) & \text{if } a,b \text{ are odd} \end{cases}$$

(b) 根据上述等式设计一个计算最大公约数的分治法。

(c) 如果a和b是n位整数，分析你的算法的时间复杂度。注意，由于n可能很大，不能假设基本算术运算（如加法）花费常数时间。