实验报告

【实验目的】

1、 通过本次实验，熟悉编程环境，为后续实验做好铺垫。
2、 回顾数论以及有限域上的基本算法，加深对其理解，为本学期密码学课程及实验课打好基础。
3、 通过本次实验，了解$GF(2^n)$上元素的性质及其四则运算。
4、 掌握上的不可约多项式的判定和生成算法。

【实验环境】

- Python 3.9 64-bit

【实验内容】

一、Euclid 算法

1、算法流程

以矩阵视角，实现以下矩阵变换

$$\begin{pmatrix} a & 1 & 0 \\ b & 0 & 1 \end{pmatrix} \xrightarrow{\text{初等行变换}} \begin{pmatrix} (a,b) & x & y \\ 0 & m & n \end{pmatrix}$$

即可得到满足$ax+by=(a,b)$的$x,y,(a,b)$。

2、测试样例及结果截图

256	2022-03-03 10:56:24	Accepted	Python	38ms	8552KB	

3、讨论与思考

- 扩展 Euclid 算法所占时间空间较 Euclid 算法更大，因为它能得到x,y，这对于求逆很方便。
- 考虑到负数的情况和结果的唯一性，对结果进行了再次处理。

二、快速幂取模

1、算法流程

欲计算$a^b \bmod p$

- 将b表示成二进制形式，令初始化结果为 1；
- 从b的最低位开始，若当前位为 1，则将当前结果乘以a模p；若当前位为 0，则当前结果不变；
- $a \leftarrow a^2 \bmod p$；
- repeat，直到b的每一位运算完毕。

2、测试样例及结果截图

181	2022-03-03 10:25:17	Accepted	Python	59ms	8548KB	

3、讨论与思考

- 相比常规的幂取模算法，快速幂模算法时空效率更高。

三、中国剩余定理

1、算法流程

- $M = m_1 m_2 m_3$，$M_i = \dfrac{M}{m_i}(i = 1, 2, 3)$；
- $x = a_1 M_1^{-1} M_1 + a_2 M_2^{-1} M_2 + a_3 M_3^{-1} M_3 \bmod (m_1 m_2 m_3)$。

2、测试样例及结果截图

3、讨论与思考

- 题目中明确说明x为最小正整数，所以还需考虑$b_1, b_2, b_3 = 0$的情况。
- 实验中调用了扩展 Euclid 算法来求模逆。

四、素性检测算法

1、算法流程

- $n - 1 = 2^k \cdot q$，随机在$(1, n-1)$选取a；
- 若$a^q \bmod n = 1$，n可能是 prime；
- 在$[0, k)$中取j，若$\exists a^{2^j q} \bmod n = n - 1$，则$n$很可能是 prime；
- 通过测试的很可能为 prime，不通过的必为 composite。

伪代码：

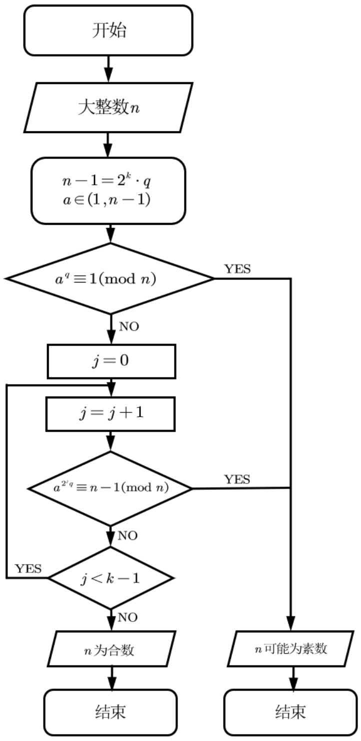
Algorithm 1: Miller Rabin 素性检测

Input: great integer n

Output: YES or NO

```
function "miller_rabin()"
    n - 1 = 2^k · q
    for i ← 1 to 10:
        a ← 2 to n - 1 的随机数
        if a^q ≡ 1(mod n):
            return YES
        for j ← 0 to k - 1:
            if a^(2^j · q) ≡ n - 1 (mod n):
                return YES
    return NO
end function
```

流程图：



2、测试样例及结果截图

413	2022-03-03 11:33:18	Accepted	Python	724ms	11732KB	

3、讨论与思考

- 该算法具有随机性，故即使程序写得不错了，结果也可能由于随机数选择的不同而不同。
- 为了再保证效率的前提下提高检测的准确性，将 a 的取值限制在了 $[n/2, n-1)$,并取了 10 个a。
- 该算法的实现效率依赖于模幂算法

五、厄拉多塞筛算法

1、算法流程

- 建立列表$l[\ ]$标记是否被筛去；
- 遍历2 to $\sqrt{n}$，如果$l[i]$没被筛去，则筛去下标为i倍数的数；
- 返回没有被筛去的数。

2、测试样例及结果截图

3717	2022-03-10 09:16:54	Accepted	Python	457ms	48680KB	

3、讨论与思考

- 由于每次都会遍历到未被筛去的倍数，我们只需遍历2 to $\sqrt{n}$ 即可；

- 为提高效率，避免一些重复的筛去，可以在初始化时去掉偶数。

六、有限域的四则运算

1、算法流程

- 加、减法即按位异或运算。
- 乘法运算利用不断加法来实现：①判断$a2$（被乘数）的最低位，为 1 则加$a1$（进行异或运算）并左移一位，否则直接左移一位；②判断$a1$的最高位，为 1 则与不可约多项式$poly$进行异或；③将$a2$右移一位，重复①。
- 除法：初始化商q为 0，当$a1$的 bit 长度大等于$a2$的 bit 长度时，$a2$左移k位后与$a1$长度相等，此时将q加上 1 左移k位后的值，将$a1$减去$a2$左移k位后的值，再比较$a1$与$a2$大小，重复上述步骤。

2、测试样例及结果截图

1343	2022-03-03 22:33:50	Accepted	Python	39ms	8488KB	

3、讨论与思考

- 该乘法算法也可以实现2^n元有限域乘法运算，只需改变不可约多项式$poly$即可。
- 对于p^n元有限域需将输入转化为p进制的形式，加、减法运算转化为模p的加减法运算。

七、有限域的快速模幂

1、算法流程

- 将k转化为二进制字符串；
- 从k的最低位开始，为 1 则将res乘以a（调用五中的乘法函数），为 0 则结果不变；
- $a \leftarrow a \times a$（调用五中的乘法函数）；
- 重复上述步骤直至k的每一位运算完毕。

2、测试样例及结果截图

1371	2022-03-03 22:58:17	Accepted	Python	36ms	8564KB	

3、讨论与思考

- 该算法与二完全一致。

八、有限域扩展 Euclid 算法

1、算法流程

与一类似，以矩阵视角不断辗转相除。

2、测试样例及结果截图

1411	2022-03-03 23:21:07	Accepted	Python	41ms	8420KB	

3、讨论与思考

- 该算法与一完全一致，但需注意其中的乘、除运算需调用五中有限域的乘、除运算。

九、有限域求逆元

1、算法流程

由 Extended Euclid 算法，可以得到：

$$x1 \times a + x2 \times b = 1$$
$$\Rightarrow a^{-1} = x1 \pmod b$$

故取出 Euclid 算法中最后得到的$x1$即可。

2、测试样例及结果截图

1995	2022-03-05 09:16:09	Accepted	Python	31ms	8512KB	

3、讨论与思考

- 该算法仍然调用了有限域的乘法与除法。

十、本原多项式的判定和生成

1、算法流程

- 以厄拉多塞筛算法的思想遍历得到 8 次不可约多项式；
- 筛去不能整除$x^{2^8-1}-1$的多项式和能整除$x^q+1(q<2^8-1)$的多项式；
- 未被筛去的即为 8 次本原多项式。

2、测试样例及结果截图

4008	2022-03-10 11:44:21	Accepted	Python	269ms	8388KB	

3、讨论与思考

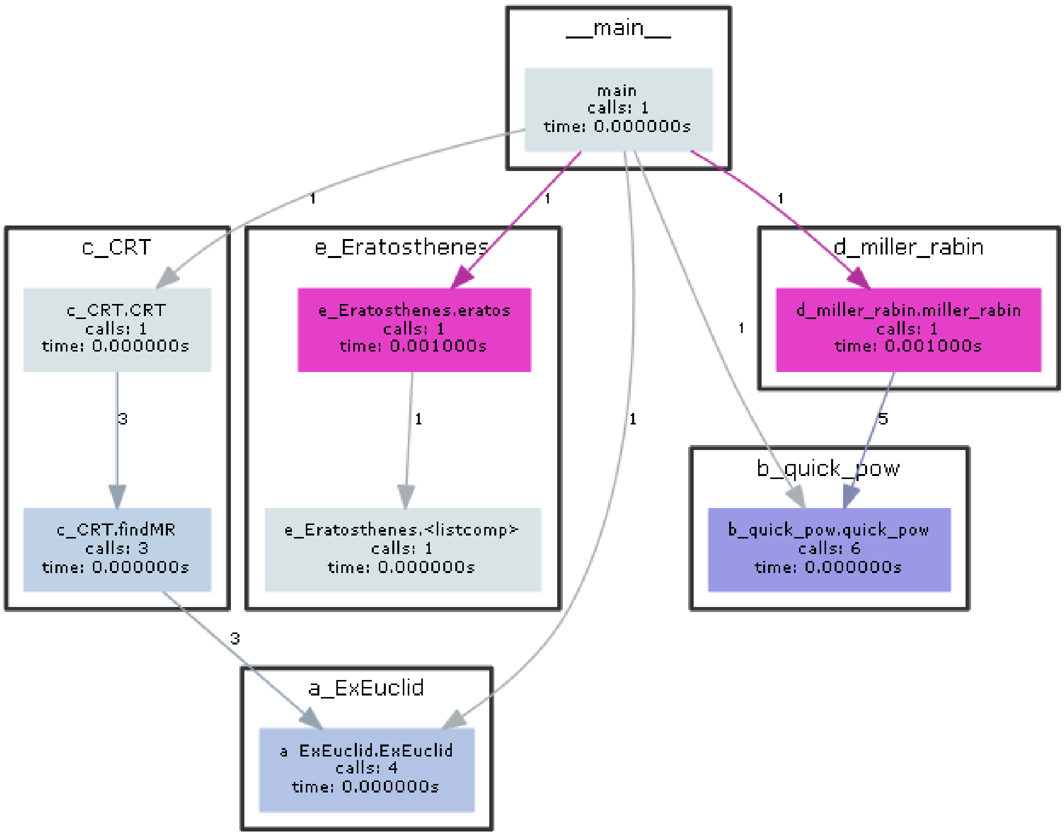
- 该算法调用了$GF(2^8)$乘法、除法的函数；
- 可以通过在程序中直接输入低次的不可约多项式（如 4 次不可约多项式）来避免不可约多项式的重复筛选。

【收获与建议】

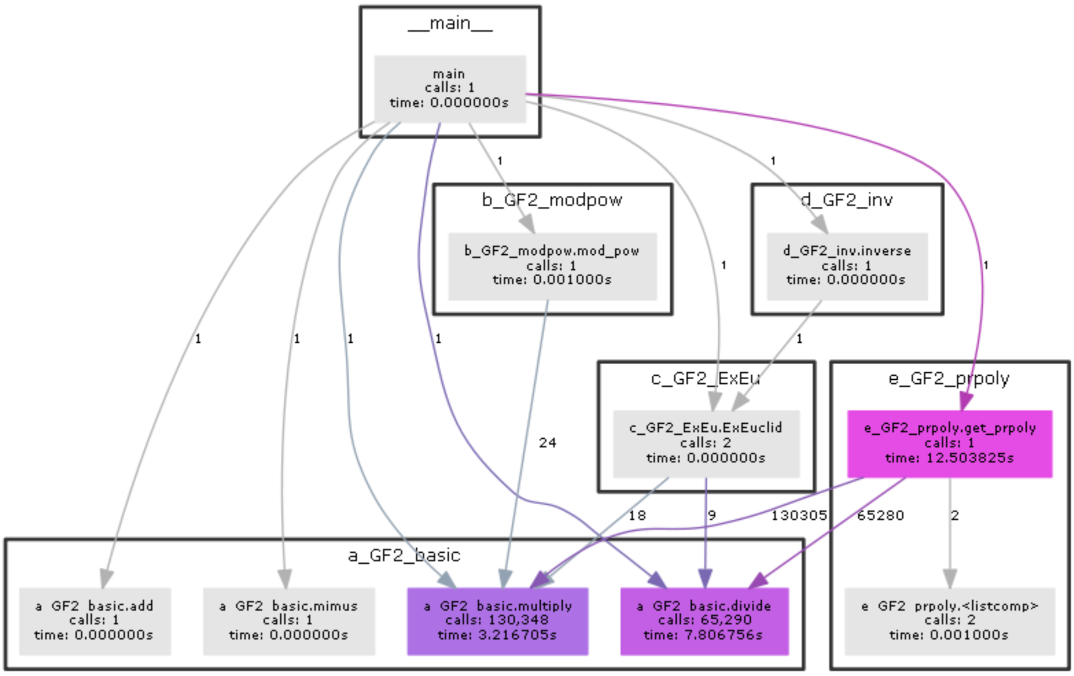
- 收获：加深了对基本算法以及有限域的理解，巩固了编程能力。
- 建议：程序提交、实验报告、程序检查三方面的工作有一定的重复性：算法流程&程序检查、程序提交&伪代码，个人建议可以适当的删减实验检查中的重复部分以减轻同学和助教们的负担。

【函数调用关系】

第一章（函数 a，b，c，d，e 分别对应算法一、二、三、四、五）



第二章（函数 a，b，c，d，e 分别对应算法六、七、八、九、十）



【思考题】

见相应部分的讨论与思考。