西安电子科技大学网信院

信息安全基础与密码学
综合实验

实 验 报 告（一）

Fermat 素性检测算法

班级：

姓名：

学号：

日期：

一、实验目的（包括实验环境、实现目标等等）

1.实验环境：Windows11，tdm64-gcc-10.3.0，VSCode

2.实验目标:给定整数 p 和安全参数 k,判断 p 是否为合数或有(1-1/2)^k 的概率为素数。

二、方案设计

1.背景：Fermat 小定理

2.原理：

- 给定素数 p,a∈Z，则有 $a^{p-1} \equiv 1 (\mathrm{mod}\ p)$
- 奇整数 m，若任取一整数 2≤a≤m-2，(a,m)= 1，使得 $a^{m-1} \equiv 1 (\mathrm{mod}\ m)$，则 m 至少有 1/2 的概率为素数

3.算法步骤：

给定奇整数 m≥3 和安全参数 k

a.随机选取整数 a，2≤a≤m-2;

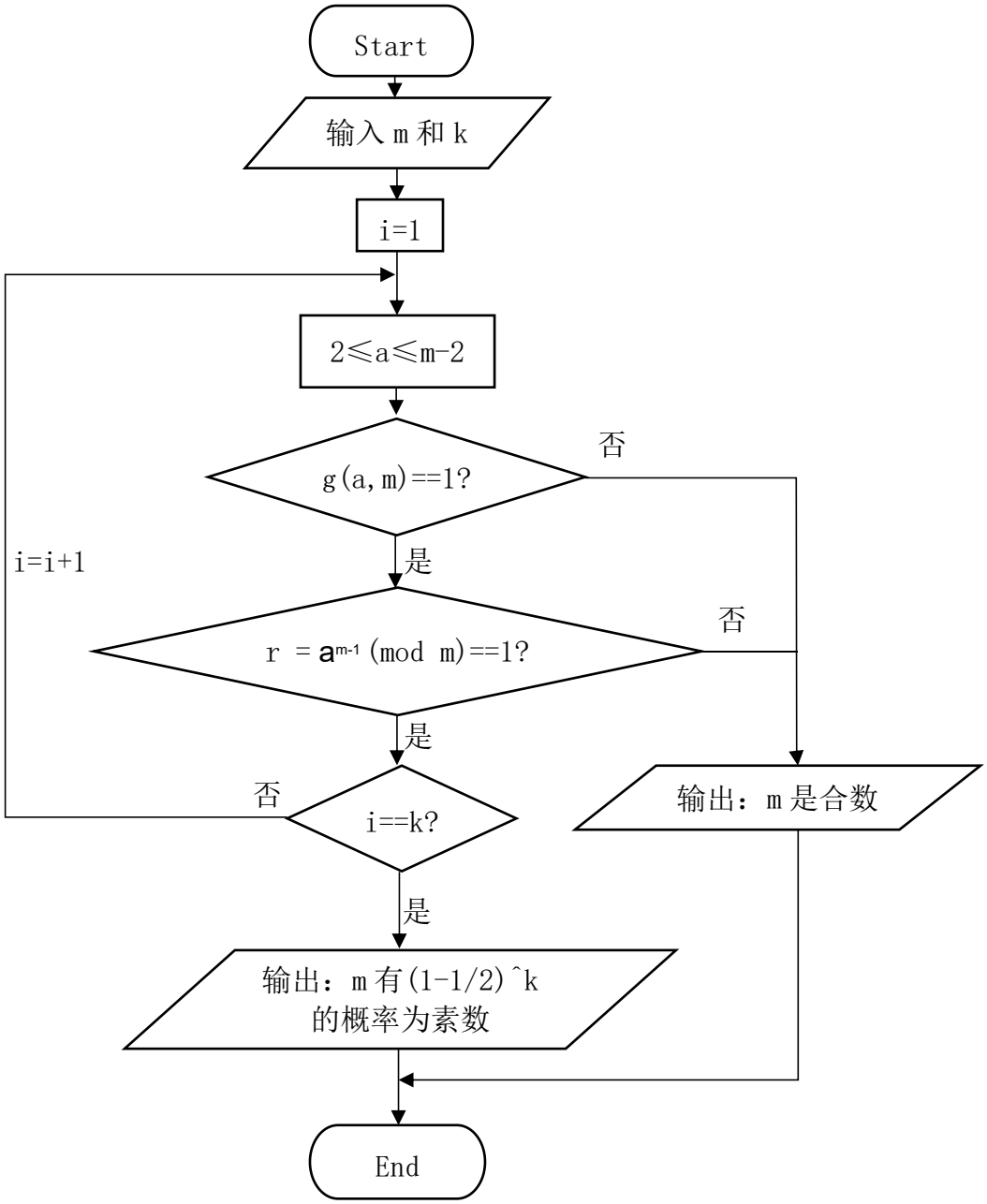
b.计算 g=(a, m)，如果 g =1，转(3)；否则跳出，m 为合数

c.计算 $r = a^{m-1} (\mathrm{mod}\ m)$，如果 r=1,m 可能是素数，转(1)；否则跳出，m 为合数

d.重复上述过程 k 次，如果每次得到 m 可能为素数，则 m 为素数的概率为 $1-(1/2)^k$

三、方案实现

1.算法流程图



2. 主要函数介绍

a. 快速模指数算法，求 a**b(mod c)

```
def quick_algorithm(a, b, c):  # 求a**b mod c
    a = a % c
    ans = 1
    while b != 0:
        if b & 1:
            ans = (ans * a) % c
        b >>= 1
        a = (a * a) % c
    return ans
```

b. 素数判断函数（判断 k 次）

```
def Isprime(m, k):  # 判断 k 次 p 是否为素数
    for i in range(k):
        a = random.randint(2, m - 2)
        if quick_algorithm(a, m - 1, p) != 1:
            return 0
        return 1
```

3. 算法实现主要代码

```
m = int(input("请输入要检测的数："))
k = int(input("请输入检测次数："))
if (Isprime(m, k)):
    probability = 1 - 0.5**k  # 计算 m 为素数的概率
    print("所检测的数是素数的概率为", probability)
else:
    print("所检测的数是合数")
```

四、数据分析

统一选择 k=4

1. 输入 m=“1.txt”

2. 输入 m=“2.txt”

```
# m = int(input("请输入要检测的数："))
with open('D:/2.txt', 'r', encoding='utf-8') as f:
    m = int(f.read())
k = int(input("请输入检测次数："))
if (Isprime(m, k)):
    probability = 1 - 0.5**k
    print("所检测的数是素数的概率为", probability)
else:
    print("所检测的数是合数")
```

问题 输出 JUPYTER 调试控制台 终端

```
PS C:\Users[redacted]
[redacted]"
请输入检测次数：4
```

3. 输入 m=“3.txt”

```
with open('D:/3.txt', 'r', encoding='utf-8') as f:
    m = int(f.read())
k = int(input("请输入检测次数："))
if (Isprime(m, k)):
    probability = 1 - 0.5**k
    print("所检测的数是素数的概率为", probability)
else:
    print("所检测的数是合数")
```

问题 输出 JUPYTER 调试控制台 终端

```
PS C:\Users[redacted]
[redacted]"
请输入检测次数：4
所检测的数是合数
```

4. 输入 m=“4.txt”

```
# m = int(input("请输入要检测的数："))
with open('D:/4.txt', 'r', encoding='utf-8') as f:
    m = int(f.read())
k = int(input("请输入检测次数："))
if (Isprime(m, k)):
    probability = 1 - 0.5**k
    print("所检测的数是素数的概率为", probability)
else:
    print("所检测的数是合数")
```

问题 输出 JUPYTER 调试控制台 终端

```
PS C:\Users[redacted]
[redacted]
请输入检测次数：4
所检测的数是合数
```

五、思考与总结

1. 如果有一个整数$\boldsymbol{a}$，$(\boldsymbol{a},\boldsymbol{m})=\boldsymbol{1}$，使得$\boldsymbol{a^{m-1}\equiv 1\ mod\ m}$ 则$\boldsymbol{m}$一定是一个素数吗？为什么？（请简述并举例说明，不能只简单回答“是”或“不是”）

   答：m 应为素数或伪素数

   例如：$m=63,(8,63)=1,8^2\equiv 1(\mathrm{mod}\ 63)$

   $8^{63-1}=8^{62}=(8^2)^{31}\equiv 1(\mathrm{mod}\ 63)$

2. Fermat 素性检测中都用到了哪些运算？分别实现什么功能？请简述。

   1.求最大公约数，确保随机数 a(2≤a≤m-2)与 m 互素，减少计算复杂度

   2.模指数运算,利用当前选定的随机数 a 检验 m 是否符合 Fermat 小定理，检验其是否为素数。若检验结果符合 Fermat 小定理，则加大 m 为素数的概率；否则 m 为合数。

3. 你还了解哪种素性检测算法？请简述，并分析其与 Fermat 素性检测算法的区别与联系。

   答：米勒-拉宾素性检验。因为伪素数的存在，所以 Fermat 素性检验不完全可靠。但是，如果以不同的 a 为底重复测试，便可降低错误率。但重复检验并不能避开卡迈克尔数（绝对伪素数），所以有时需要一个更加完整的素性测试。

   米勒-拉宾素性检验是目前最被认可的素性检验。它在 Fermat 素性检验的基础上又添加了另外的判断条件，令卡迈克尔数更难“骗”住它。但是，它和 Fermat 素性检验一样只能判断一个数字是合数还是可能是素数。

4. 实验过程中还遇到了什么问题，如何解决的？通过该实验有何收获？

   答：VSCode 终端不允许直接粘贴测试用例，open(“1.txt”)得到的类型为 io 对象，不知道如何转换为 int 格式，最后通过搜索解决问题。

   本次实验我学会了快速模指数算法，熟练了编程，加深了我对 Fermat 小定理的理解。