实验项目名称： 快速指数和M-R算法 实验学时： 4

同组学生姓名： 无 实验地点： 科技楼4-1204

实验日期： 2023.11.29 实验成绩：

批改教师： 黄丹丹 批改时间： 2023.12.06

实验5 快速指数和M-R算法

一、基于OBE模式的实验目的和要求

1、掌握快速指数算法，并会运用其实现模幂运算；

2、掌握Miller-Rabin算法，并会运用其检测整数的素性。

二、实验仪器和设备

Visual C/C++

三、实验过程

1、设计实现算法程序，利用快速指数算法求*am*(mod *n*)。

FastExponent (*a*, *m*, *n*)



1）d = 1;

2）for *i* = k－1 downto 0 do

{

d← d2mod *n* ；

if b*i* =1 then d ← (d×*a* )mod *n*

}

return d.

测试示例：当 *a* = 160, m = 77, n = 221时，求出*am*(mod *n*)。

2、设计实现算法程序Miller-Rabin素性检测算法。

WITNESS (*a*，*n*)

1）将*n*-1表示为二进制形式 ；

2）d = 1;

for *i* = k downto 0 do

{

x ← d;

d ← d2mod *n* ；

if d = 1 and (x ≠ 1) and (x ≠ n-1) then return *FALSE*;

if b*i* = 1 then d ← (d×*a* )mod *n*；

}

if d ≠ 1 then return *FALSE*;

return *TRUE*.

测试示例：1）当*n* = 3337时，随机选择参数*a* （5个左右），使得算法输出为FALSE，因此输出*n*不是素数；2）当 *n* = 2333时，随机选择参数*a*（至少10个），使得算法输出都是TRUE，因此说明*n* 有很大概率是素数，否则是强伪素数。例如，当10个参数测试通过时*n*是素数的概率至少为  > 99.999%。

3、程序代码：

#include <stdio.h>#include <stdlib.h>int FastExponent(int a, int m, int n) {    int d = 1;    int binaryM[16] = {0};    int i = 0;  
    while (m > 0) {        binaryM[i] = m % 2;        m = m / 2;        i++;    }      
    for (int j = i - 1; j >= 0; j--) {        d = (d \* d) % n;        if (binaryM[j] == 1) {            d = (d \* a) % n;        }    }    return d;}int WITNESS(int a,int n) {    if (n <= 1)   
        return 0;   
      
    if (n == 2 || n == 3)        return 1;      
    if (n % 2 == 0)        return 0;   
      
    int d = n-1;    int k = 0;      
    while (d % 2 == 0) {        d = d / 2;        k++;    }      
    int x, witness;    for (int i = 0; i < a; i++) {        witness = (rand() % (n - 3)) + 2;   
        x = FastExponent(witness, d, n);        if (x == 1 || x == n - 1) {            continue;    
        }          
        for (int j = 0; j < k - 1; j++) {            x = FastExponent(x, 2, n);              
            if (x == n - 1) {                break;    
            }              
            if (x == 1) {                return 0;   
            }        }          
        if (x != n - 1) {            return 0;        }    }      
    return 1;}int main() {    int n;    printf("输入需要验证的数字：");    scanf("%d",&n);    if (WITNESS (10, n)) {        printf("%d 有很大可能是素数\n", n);    }else{        printf("%d 不是素数\n", n);    }      
    return 0;}

四、实验结果与分析

测试结果截图与分析如下：

文本

描述已自动生成

文本

描述已自动生成

首先调用快速指数算法计算数字较大的次幂运算，这种算法的好处是每次结果都会小于n，输出结果如图。

文本

描述已自动生成

文本

描述已自动生成

在调用M-R算法，M-R算法里需要用到快速指数算法。判断次数选择10，这精确率已经达到99.999%，这几乎就是100%，但是数学上需要严谨，所以输出“很大可能”，但是对于密码学来说这个精度已经够用了。

五、基于OBE模式的学生自我评价与体会

（该实验对自身分析、设计、思辨、创新等个人综合能力与素质的影响）

通过本次实验我学习到了快速指数算法和M-R算法，快速指数算法能在一定程度上解决次幂运算结果太大不方便继续模运算，快速指数算法就是将次幂转化成二进制形式，根据二进制位数按照平方或乘法运算规则进行计算。通过这种算法可以将每次运算结果都控制在小于模数的范围内，可以大大减小计算量。M-R算法是借助快速指数算法进行大素数判断。虽然不是100%，但是对于判断次数为10来说已经很精确了。总之，通过本次实验，我不止学到了新的算法用于处理较大数据，还锻炼了C语言编程能力，让我受益匪浅。