密码学第三次实验报告

一．实验目的：

请提交一个满足 RSA 算法要求的模数 N，要求：使用一台 PC 机运行 Sage（或相类似软件）对 N 进行分解，所需时间大概为 60 分钟。误差可在正负 5 分钟之内。实验报告中请描述选择 N 的依据，生成的方法，及相关的检测数据。

二．实验过程：

（1）模数 N 的生成，要生成模数 N，先随机找出两个大素数 p 与 q，通过米勒拉宾算法检验 p 与 q 是否为素数，并根据算法生成特定位数的随机数，从而得到特定位数的大素数。

（2）米勒拉宾算法的实现是使用 python 程序设计语言实现，原因是 python 该语言的大数据系统可以方便地解决大数之间的运算。

（3）把生成的 N 放到 Sage 上运行，直接在 Sage 的官网进行在线测试，使用 factor 函数对 N 进行分解并计算时间，结果分析 p 和 q 为 134bits 左右时，即 N 大概为 268bits 的时候需要耗费一个小时进行对 N 的分解。

三. 米勒拉宾算法代码：

```
import random
# (a * b) % c
def mod_pro(a, b, n):
  return a*b % n

# a ^ b % c
def mod(a, b, c):
  if b == 1:
    return a % c
  a %= c
  ret = 1
  while b:
    if b&1:
      ret = mod_pro(ret,a,c)
    a = mod_pro(a,a,c)
    b >>= 1
  return ret

def check (a, n, m, k):
    ret = mod(a, m, n)
    last = ret
    for i in range(0, k):
```

```
        ret = mod_pro(ret, ret, n);
        if (ret == 1 and last != 1 and last != n-1): return True
        last = ret
    if (ret != 1): return True
    return False

def is_prime(n, t):
  if n < 2:
    return False
  if n == 2:
    return True
  k = 0
  m = n - 1
  while m&1 == 0:
    m >>=1
    k += 1
  for i in range(0, t):
    a = random.randint(0, n)
    if(check(a,n,m,k)):
      return False
  return True

l = input()

while 1:
  p = random.randint(2**l, 2**(l + 1))
  if is_prime(p,20):
    print p
    break

while 1:
  q = random.randint(2**l, 2**(l + 1))
  if is_prime(q,20):
    print q
    break

print p * q
```