Московский авиационный институт
(государственный технический университет)
Факультет прикладной математики и физики
Кафедра вычислительной математики и программирования

Лабораторная работа №1

по спецкурсу «Криптография»:
Генерация простых чисел

Выполнил:	Карпова В.А.
Группа:	08-306
№ по списку:	9
Преподаватель:	Рисенберг Д.В.
Оценка:	
Дата:	

Москва
2012 г.

Постановка задачи.

Необходимо написать программу на языке C++, C# или Python, реализующую алгоритм проверки на простоту и генерации простых чисел.

Вариант 4: Проверка числа на простоту с использование полного разложения n-1 на простые множители с нахождением образующей (сертификат Пратта).

Сертификат Пратта. Алгоритм.

Вероятностный алгоритм Пратта базируется на следующих предположениях:
Для каждого целого n > 1 следующие 3 утверждения эквивалентны:

1) Порядок группы по умножению из n элементов равен n-1. $|Z_n^*| = n-1$.
2) Целое число n - простое.
3) Элемент a, принадлежащий группе по умножению, т.ч. $a^{n-1} \equiv 1 \; mod \; n$ и для каждого простого делителя q числа n-1, $a^{\frac{n-1}{q}} \not\equiv 1 \; mod \; n$.

Алгоритм Пратта работает следующим образом:
Принимает на вход простое число и выдает доказательство (сертификат) того, что оно действительно является простым.
Для этого используются следующие шаги:
1) Находим элемент группы n порядка n-1. aϵZ_n^*, $|Z_n^*| = n-1$.
2) Определяем факторизацию p-1.$n-1 = \prod_{i=1}^{k} q_i^{\alpha_i}$
3) Доказываем, что n простое. Для этого доказываем, что a - образующая рассматриваемой группы Z_n^*. Для образующей проверяется условие $a^{n-1} \equiv 1 \; mod \; n.$ А для каждого простого q_iпроверяем: $a^{\frac{n-1}{q_i}} \not\equiv 1 \; mod \; n.$
4) Рекурсивно показываем, что каждое q_i простое, $1 \le i \le k$.

Генерация большого простого числа.

Алгоритм создания большого простого числа использует следствие из теоремы Ферма и уже реализованную проверку на простоту.
Теорема Ферма: Пусть N,S -- нечетные натуральные числа, N-1 = S*R, причем для каждого простого делителя q числа S существует целое число a, т.ч. $a^{N-1} \equiv 1 \; mod \; N,$ $GCD\left(a^{\frac{N-1}{q}} - 1, N\right) = 1$. Тогда каждый простой делитель p числа N удовлетворяет сравнению p = 1(mod 2S).
Следствие: Если выполнены условия теоремы Ферма и R≤4*S+2, то N -- простое число.

Алгоритм:
Берем четное случайное число в диапазоне от (S;4*S+2). На простоту будем проверять число N = R*S+1. Найдем образующую с помощью сертификата Пратта и применим условия теоремы Ферма. Получим ответ и число итераций, за которое было найдено простое число.

Оценка сложности.

Сертификатом Пратта для простого натурального числа n называется набор {p1,..,pk,a}. Если сертификат Пратта известен, то с его помощью можно убедиться в простоте n за O($\log n^2$).
Нахождение сертификата Пратта не такая простая задача. Если число n -- простое, то количество образующих у соответствующей группы = $\frac{n}{loglogn}$. С условием того, что факторизация уже задана -- получаем сложность O($loglogn$).

Реализация.

```
def isprime(n,p):
      if n==2: return True
      k = len(p)
      kn = k
      for i in range(0,round(n*log(log(n)))):
            a_i = randint(2,n-1) # находим образующую группы
            check1 = powmod(a_i,n-1,n) #проверка 1го условия
            if (check1!=1):
                  return False

            # для всех простых делителей показываем выполнимость 2го условия
            while k>0:
                  #~ print(a_i)
                  check2 = powmod(a_i,(n-1)//p[k-1],n)
                  if (check2!=1):
                        k -= 1
                  else: break
            if (k==0):
                  print("a = ",a_i)
                  return True
            else : kn -= 1

      return False

def gen_prime(S):
      lp = [2]
      print ("S = ",S)
      iters = 0
      while True :
            iters += 1
            R = randint(S,4*S+2)
            # even R
            if (R%2==1): R+=1
            N = R*S+1
            if isprime(N,lp): # and check_Ferma(N,R):
                  print ("\nprime : ", N, "iters = ", iters)
                  break
            #~ print ("not prime", N, end = ";")
      return N
```

Выводы.

Простые числа широко используются в практике. Например, в качестве ключей шифрования. Также, простые числа могут быть использованы для построения генераторов псевдослучайных чисел.

Однако, определение простоты заданного числа в общем случае не такая уж тривиальная задача. Только совсем недавно было доказано, что она полиномиально разрешима.

А с помощью сертификата Пратта можно ответить, является ли число простым за полиномиальное время.