Математические основы защиты информации
Лабораторная работа №2
Генерация простых чисел

БГУ, ММФ, Каф. ДУиСА,
доцент Чергинец Д.Н.

Метод пробных делений

Какие же есть способы проверки числа на простоту?

Наиболее древний алгоритм нахождения простых чисел — решето Эратосфена (III в. до н.э.) — позволяет выписать все простые числа, не превосходящие данного числа n, а также найти наименьший простой делитель числа n, если n — составное число. Алгоритм заключается в следующем: записываем последовательно все числа от 2 до n, затем в полученной таблице вычеркиваем каждое второе число после 2, каждое третье после 3, каждое пятое после 5 и т.д. При этом каждый раз считаются и уже вычеркнутые числа. После каждой процедуры вычеркивания первое, оставшееся не вычеркнутым, число k является простым, а затем вычеркиваются все числа, следующие за k и кратные k. Вычеркивания производят до тех пор, пока $k \le \sqrt{n}$, так как, очевидно, любое составное число a имеет делитель $\le \sqrt{a}$.

Аналогичен решету Эратосфена метод пробных делений. Необходимо делить n с остатком на числа d, $1 < d \le \sqrt{n}$, если найдется такое d, что $d \mid n$, то n — составное, иначе — простое. Однако этот алгоритм имеет экспоненциальную сложность: если n имеет длину $N = \lceil \log_2 n \rceil + 1$, т.е. n в двоичной записи содержит N цифр, то надо проделать порядка $\sqrt{n} = O\left(2^{\frac{N}{2}}\right)$ операций деления с остатком, т.е. данный алгоритм является

экспоненциальным. Поэтому для чисел с большими делителями метод пробных делений неприменим.

Задание 1.

При помощи метода пробных делений выяснить, является ли число **n** простым, найти время вычисления при помощи функции Timing.

Вариант 1.

n = 100 100 000 003

Вариант 2.

n = 100 200 000 013

Вариант 3.

n = 100 300 000 027

Вариант 4.

n = 100 400 000 087

Вариант 5.

n = 100 500 000 109

Вариант 6.

Вариант 7.

Вариант 8.

Вариант 9.

Вариант 10.

Вариант 11.

Вариант 12.

Вариант 13.

Вариант 14.

Вариант 15.

Вариант 16.

Вариант 17.

Вариант 18.

Вариант 19.

Вариант 20.

Вариант 21.

Вариант 22.

Вариант 23.

Вариант 24.

Вариант 25.

Вариант 26.

Вариант 27.

Вариант 28.

Вариант 29.

Вариант 30.

Малая теорема Ферма

Видим, что для генерации больших простых чисел, которые мы будем использовать в шифрах RSA и Эль-Гамаля, Метод пробных делений не подходит.
Ранее мы познакомились с теоремой Эйлера (в случае, когда модуль **n** - простое число, теорема Эйлера является малой теоремой Ферма), из которой следует, что
если **n** — простое число, то $a^{n-1} \equiv 1 \pmod{n}$ для всех **a** (**1 < a < n**).
Таким образом, если мы найдем такое число **a** (**1 < a < n**), что $a^{n-1} \not\equiv 1 \pmod{n}$, то **n** — составное. Если $p_1 > 1$ является делителем **n**, то $p_1^{n-1} \not\equiv 1 \pmod{n}$.
Следовательно, для каждого составного **n** существует такое число **a**, 1 < **a** < **n**, что $a^{n-1} \not\equiv 1 \pmod{n}$.

Задание 2.

При помощи малой теоремы Ферма доказать, что число **n** составное.
Попробуйте разложить число **n** на простые множители при помощи

встроенной функции **TimeConstrained[FactorInteger[n],60]**.

Вариант 1.

```
n =
 68 904 949 027 325 924 751 224 827 790 018 638 409 784 780 132 917 815 689 952 136 862 340 162 329 ×
  903 379 072 395 133
```

Вариант 2.

```
n =
 196 671 404 219 308 171 101 107 973 342 594 626 980 333 240 682 660 328 298 258 773 792 459 126 666 ×
  611 318 681 327 929
```

Вариант 3.

```
n =
 337 827 555 033 639 120 756 073 895 514 502 653 459 962 409 952 790 154 678 124 658 029 894 911 975 ×
  819 575 037 922 823
```

Вариант 4.

```
n =
 550 217 706 178 950 470 618 593 125 211 520 099 184 513 268 448 919 800 841 485 433 152 058 379 489 ×
  851 223 001 174 161
```

Вариант 5.

Вариант 6.

Вариант 7.

Вариант 8.

Вариант 9.

Вариант 10.

Вариант 11.

Вариант 12.

Вариант 13.

Вариант 14.

Вариант 15.

Вариант 16.

Вариант 17.

Вариант 18.

Вариант 19.

Вариант 20.

Вариант 21.

Вариант 22.

Вариант 23.

Вариант 24.

Вариант 25.

Вариант 26.

Вариант 27.

Вариант 28.

Вариант 29.

Вариант 30.

Числа Кармайкла

Видим, что для составных чисел **n** обычно уже при **a = 2** выполняется условие $\mathbf{a^{n-1} \not\equiv 1 \pmod n}$,
что доказывает их не простоту. Но если нам попалось простое число,
то данный метод ещё хуже метода пробных делений,
так как для того чтобы доказать простоту числа **n**,
необходимо доказать, что $\mathbf{a^{n-1} \equiv 1 \pmod n}$ для всех **a**, $\mathbf{1 < a \leq \sqrt{n}}$.

Более того, есть следующие числа

Определение.
Если для составного числа **n** выполняется $\mathbf{a^{n-1} \equiv 1 \pmod n}$ для всех **a**, $\mathbf{1 < a < n,\ (a,\ n) = 1}$, то число **n** называется *числом Кармайкла*.

Наименьшим числом Кармайкла
является 561. Чисел Кармайкла бесконечно много.

```
n = 561;
PowerMod[2, n - 1, n]
PowerMod[3, n - 1, n]
PowerMod[4, n - 1, n]
PowerMod[5, n - 1, n]
FactorInteger[n]
```

```
1

375

1

1

{{3, 1}, {11, 1}, {17, 1}}
```

Вероятностный тест на простоту Миллера-Рабина

Пусть дано число **n,**
которое мы будем исследовать на простоту. Если **n** делится на **2**,
то оно составное,
поэтому будем считать его нечетным. Тогда $\mathbf{n-1=2^s t}$,
где **t** — нечетно, $\mathbf{s>0}$.
Для всякого натурального **a** из промежутка $\mathbf{1<a<n}$ имеем равенства

$$a^{n-1}-1=a^{2^s t}-1=\left(a^{2^{s-1}t}+1\right)\left(a^{2^{s-1}t}-1\right)=\left(a^{2^{s-1}t}+1\right)\left(a^{2^{s-2}t}+1\right)\left(a^{2^{s-2}t}-1\right)=$$
$$\left(a^{2^{s-1}t}+1\right)\left(a^{2^{s-2}t}+1\right)\ldots\left(a^{t}+1\right)\left(a^{t}-1\right) \qquad (1)$$

Если число **n** – простое, то $\mathbf{a^{n-1}-1\equiv 0\ (mod\ n)}$,
а так как $\mathbf{Z_n}$ — поле, то в нем нет делителей нуля и в правой
части равенства (1) обязательно одна из скобок равна нулю,
то есть выполняется хотя бы одно из сравнений

$$a^{2^i t}\equiv n-1\ (\mathrm{mod}\ n),\quad i:=0,\ \ldots,\ s-1, \qquad a^t\equiv 1\ \ (\mathrm{mod}\ n). \qquad (2)$$

Определение. Натуральное число **a**, $1<\mathbf{a}<\mathbf{n}$, называется свидетелем простоты числа **n,** если выполняются два условия

1. НОД(a, n) = 1;
2. Справедливо хотя бы одно из сравнений (2).

Очевидно, что если **n** простое, то каждое число **a,** $\mathbf{1<a<n,}$ является

свидетелем простоты числа **n**. Составное же число **n** имеет не более $\frac{n-1}{4}$ свидетелей простоты. Если число ***a***, **1 < *a* < *n*,** не является свидетелем простоты числа ***n*,** то ***n*** — составное.

Задание 3.

Написать функцию **WitnessQ[a_Integer, n_Integer]**, которая возвращает **True**, если число **a** является свидетелем простоты числа **n** и **False** в противном случае.

Задание 4.

Реализовать вероятностный тест на простоту Миллера-Рабина, который для числа ***n*** возвращает **True**, если ***k*** наугад взятых чисел ***a*** оказались свидетелями простоты числа ***n***. Сравнить скорость вычислений со встроенной функцией PrimeQ. Является ли тест Миллера-Рабина:
- детерминированным алгоритмом?
- вероятностным алгоритмом?
- полиномиальным алгоритмом?

Задание 5.

Используя вероятностный тест на простоту Миллера-Рабина, написать функцию **PrimeGeneration[b_Integer]**, которая возвращает случайное простое число, состоящее из ***b*** бит.

```
b = 1024;
(p1 = PrimeGeneration[b]) // Timing
IntegerDigits[p1, 2] // Length
(p2 = RandomPrime[{2^(b-1), 2^b - 1}]) // Timing
IntegerDigits[p1, 2] // Length
```

```
{0.234002,
 168 876 764 814 049 528 803 594 311 767 309 140 286 497 328 899 551 553 668 494 586 714 437 970 516 ∖
  584 842 108 740 255 292 840 301 363 375 387 287 771 399 447 458 898 164 271 687 372 385 211 503 797 ∖
  597 407 209 148 737 291 391 366 056 254 816 254 973 997 237 466 363 081 044 613 375 319 841 473 080 ∖
  501 980 392 221 997 332 551 181 450 449 588 272 583 816 203 236 723 414 095 144 492 272 276 207 838 ∖
  722 914 033}
```

```
1024
```

```
{0.0780005,
 160 500 259 983 301 233 168 565 143 496 423 269 212 706 979 804 280 595 992 554 418 011 663 950 026 ∖
  530 924 327 486 241 500 019 644 429 949 310 162 576 116 864 108 931 705 802 247 179 360 025 467 100 ∖
  041 016 534 700 295 452 532 301 186 304 592 908 713 287 615 749 136 430 915 528 579 145 553 842 505 ∖
  122 990 891 627 244 389 488 118 613 955 699 046 728 572 799 477 971 408 262 785 998 392 164 391 654 ∖
  190 001 019}
```

```
1024
```

Наибольшим известным простым числом является $2^{57\,885\,161} - 1$. За нахождение простых чисел, состоящих из более чем 1 миллиарда десятичных цифр Фонд электронных рубежей (Electronic Frontier Foundation) назначил приз в 250 тыс. долларов.

Построение простых чисел

Теорема Диемитко.

Пусть $n = 2\,r\,q + 1$, где q – нечетное простое число, $r \in \mathbb{N}$, $r \le 2\,q + 1$.

Если для некоторого $a \in \mathbb{N}$, $1 < a < n$,

$a^{n-1} \equiv 1 \pmod{n}$, $\quad a^{2\,r} \not\equiv 1 \pmod{n}$,

то n – простое.

Задание 6.

На основе теоремы Диемитко написать алгоритм, который, начиная с малого простого числа, строит случайное простое число, большее x∈ℕ. Является ли полученный алгоритм:

- детерминированным алгоритмом?
- вероятностным алгоритмом?
- полиномиальным алгоритмом?

Задание 7.

Провести сравнительный анализ теста Миллера - Рабина с алгоритмом, основанном на теореме Диемитко.