John von Neumann:

Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin. For, as has been pointed out several times, there is no such thing as a random number — there are only methods to produce random numbers, and a strict arithmetic procedure of course is not such a method.

Random vs. pseudorandom behaviour

**Random behavior** -- Typically, its outcome is unpredictable and the parameters of the generating process cannot be determined by any known method. Examples:
- Parity of number of passengers in a coach in rush hour.
- Weight of a book on a shelf in grams modulo 10.
- Direction of movement of a particular $N_2$ molecule in the air in a quiet room.

**Pseudo-random** -- Deterministic formula,
- Local unpredictability, "output looks like random",
- Statistical tests might reveal more or less "random behaviour"

**Pseudorandom integer generator**
A pseudo-random integer generator is an algorithm which produces a sequence
\[ \{x_n\} = x_0, x_1, x_2, ... \]
of non-negative integers, which manifest pseudo-random behaviour.
Pseudorandom number generator

Pseudorandom integer generator

Two important statistical properties:

- Uniformity
- Independence

Random number in a interval \([a, b]\) must be independently drawn from a uniform distribution with probability density function:

\[
f(x) = \begin{cases} \frac{1}{b - a + 1} & x \in [a, b] \\ 0 & \text{elsewhere} \end{cases}
\]

Good generator

- Uniform distribution over large range of values:
  Interval \([a, b]\) is long, period = \(b - a + 1\), generates all integers in \([a, b]\).
- Speed
  Simple generation formula.
  Modulus (if possible) equal to a power of two – fast bit operations.
Random floating point number generator

Task 1: Generate (pseudo) random integer values from an interval \([a, b]\).
Task 2: Generate (pseudo) random floating point values from interval \([0,1]\).

Use the solution of Task 1 to produce the solution of Task 2.
Let \(\{x_n\}\) be the sequence of values generated in Task 1.
Consider a sequence \(\{y_n\} = \{(x_n - a) / (b - a + 1)\}\).

Each value of \(\{y_n\}\) belongs to \([0,1]\).
"Random" real numbers are thus approximated by "random" fractions.
Large length of \([a, b]\) guarantees sufficiently dense division of \([0,1]\).

Example 1
\([a, b] = [0, 1024]\).
\(\{x_n\} = \{712, 84, 233, 269, 810, 944, \ldots\}\)
\(\{y_n\} = \{712/1023, 84/1023, 233/1023, 269/1023, 810/1023, 944/1023, \ldots\}\)
\(= \{0.696, 0.082, 0.228, 0.263, 0.792, 0.923, \ldots\}\)
Linear Congruential Generator

Linear congruential generator

Linear congruential generator produces a sequence \( \{x_n\} \) defined by relations

\[
0 \leq x_0 < M,
\]

\[
x_{n+1} = (Ax_n + C) \mod M, \quad n \geq 0.
\]

Modulus \( M \), seed \( x_0 \), multiplier and increment \( A, C \).

Example 2

\[
M = 18, \quad A = 7, \quad C = 5.
\]

\[
x_0 = 4,
\]

\[
x_{n+1} = (7x_n + 5) \mod 18, \quad n \geq 0.
\]

\[
\{x_n\} = 4, 15, 2, 1, 12, 17, 16, 9, 14, 13, 6, 11, 10, 3, 8, 7, 0, 5, 4, 15, 2, 1, 12, 17, 16, \ldots
\]

sequence period, length = 18
Example 3

\[ M = 15, A = 11, C = 6. \]

\[ x_0 = 8, \]
\[ x_{n+1} = (11x_n + 6) \text{ mod } 15, \quad n \geq 0. \]

\{x_n\} = 8, 14, 5, 11, 2, 8, 14, 5, 11, 2, 8, 14, \ldots

sequence period, length = 5

Example 4

\[ M = 13, A = 5, C = 11. \]

\[ x_0 = 7, \]
\[ x_{n+1} = (5x_n + 11) \text{ mod } 13, \quad n \geq 0. \]

\{x_n\} = 7, 7, 7, 7, \ldots

sequence period, length = 1
Misconception
Prime numbers are "more random" than composite numbers, therefore using prime numbers in a generator improves randomness.
Counterexample: Example 4, all parameters are primes:

\[ x_0 = 7, \quad x_{n+1} = (5x_n + 11) \mod 13. \]

Maximum period length

Hull-Dobell Theorem:
The length of period is maximum, i.e. equal to \( M \), iff conditions 1. - 3. hold:
1. \( C \) and \( M \) are coprimes.
2. \( A-1 \) is divisible by each prime factor of \( M \).
3. If 4 divides \( M \) then also 4 divides \( A-1 \).

Example 5

1. \( M = 18, \ A = 7, \ C = 6. \) Condition 1. violated
2. \( M = 20, \ A = 17, \ C = 7. \) Condition 2. violated
3. \( M = 17, \ A = 7, \ C = 6. \) Condition 2. violated
4. \( M = 20, \ A = 11, \ C = 7. \) Condition 3. violated
5. \( M = 18, \ A = 7, \ C = 5. \) All three conditions hold
Linear Congruential Generator

Randomness issues

Example 6

\[ x_0 = 4, \]
\[ x_{n+1} = (7x_n + 5) \mod 18, \quad n \geq 0. \]

\[ \{x_n\} = 4, 15, 2, 1, 12, 17, 16, 9, 14, 13, 6, 11, 10, 3, 8, 7, 0, 5, 4, 15, 2, 1, 12, 17, 16, \ldots \]

sequence period, length = 18

\[ \{x_n \mod 2\} = 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, \ldots \]

\[ \{x_n \mod 3\} = 1, 0, 2, 1, 0, 2, 1, 0, 2, 1, 0, 2, 1, 0, 2, 1, 0, 2, 1, 0, 2, 1, 0, 2, 1, \ldots \]

\[ \{x_n \div 4\} = 0, 3, 0, 0, 3, 4, 4, 2, 3, 3, 1, 2, 2, 0, 2, 1, 0, 1, 0, 3, 0, 0, 3, 4, 4, \ldots \]

Trouble

Low order bits of values generated by LCG exhibit significant lack of randomness.

Remedy

Disregard the lower bits in the output (not in the generation process!).

Output the sequence \( \{y_n\} = \{x_n \div 2^H\} \), where \( H \geq \frac{1}{4} \log_2(M) \).
## Linear Congruential Generator

Examples of LCGs in common use

<table>
<thead>
<tr>
<th>Source</th>
<th>modulus $m$</th>
<th>multiplier $a$</th>
<th>increment $c$</th>
<th>output bits of seed in $\operatorname{rand}()$ or $\operatorname{Random}(L)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numerical Recipes</td>
<td>$2^{32}$</td>
<td>1664525</td>
<td>1013904223</td>
<td>bits 30..16 in $\operatorname{rand}()$, 30..0 in $\operatorname{irand}()$</td>
</tr>
<tr>
<td>Borland C/C++</td>
<td>$2^{32}$</td>
<td>22695477</td>
<td>1</td>
<td>bits 30.0</td>
</tr>
<tr>
<td>glibc (used by GCC)[15]</td>
<td>$2^{31}$</td>
<td>110351245</td>
<td>12345</td>
<td>bits 30..16</td>
</tr>
<tr>
<td>Borland Delphi, Virtual Pascal</td>
<td>$2^{32}$</td>
<td>134775813</td>
<td>1</td>
<td>bits 63..32 of $(\operatorname{seed} \times L)$</td>
</tr>
<tr>
<td>Turbo Pascal</td>
<td>$2^{32}$</td>
<td>134775813 (8088405&lt;sub&gt;u&lt;/sub&gt;)</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Microsoft Visual/Quick C/C++</td>
<td>$2^{32}$</td>
<td>214013 (343FD&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>2531011 (269EC3&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>bits 30..16</td>
</tr>
<tr>
<td>Microsoft Visual Basic (6 and earlier)&lt;sup&gt;[18]&lt;/sup&gt;</td>
<td>$2^{24}$</td>
<td>1140671485 (43FD43FD&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>12820163 (C39EC3&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>bits 30..16</td>
</tr>
<tr>
<td>RtUniform from Native API&lt;sup&gt;[19]&lt;/sup&gt;</td>
<td>$2^{31} - 1$</td>
<td>2147483629 (7FFFFFFED&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>2147483587 (7FFFFFFC3&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>bits 30..16</td>
</tr>
<tr>
<td>Apple CarbonLib, C++11's minstd_rand&lt;sup&gt;[20]&lt;/sup&gt;</td>
<td>$2^{31} - 1$</td>
<td>16807</td>
<td>0</td>
<td>see MINSTD</td>
</tr>
<tr>
<td>C++11's minstd_rand&lt;sup&gt;[20]&lt;/sup&gt;</td>
<td>$2^{31} - 1$</td>
<td>48271</td>
<td>0</td>
<td>see MINSTD</td>
</tr>
<tr>
<td>MMIX by Donald Knuth</td>
<td>$2^{64}$</td>
<td>636413622386793005</td>
<td>144269504088963407</td>
<td>bits 63.32</td>
</tr>
<tr>
<td>Newlib, Musl</td>
<td>$2^{64}$</td>
<td>636413622386793005</td>
<td>1</td>
<td>bits 63.32</td>
</tr>
<tr>
<td>VMS's MTHSRANDOM,&lt;sup&gt;[21]&lt;/sup&gt; old versions of glibc</td>
<td>$2^{32}$</td>
<td>69069 (10DCD&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>random8&lt;sup&gt;[22][23][24][25][26]&lt;/sup&gt;</td>
<td>$134456 = 2^{27}$</td>
<td>8121</td>
<td>28411</td>
<td>$\frac{X_n}{134456}$</td>
</tr>
<tr>
<td>POSIX&lt;sup&gt;[27]&lt;/sup&gt; [jm]rand48, glibc [jm]rand48[_r]</td>
<td>$2^{48}$</td>
<td>25214903917 (5DEEE66D&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>11</td>
<td>bits 47..15</td>
</tr>
<tr>
<td>POSIX [de]rand48, glibc [de]rand48[_r]</td>
<td>$2^{48}$</td>
<td>25214903917 (5DEEE66D&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>11</td>
<td>bits 47.0</td>
</tr>
<tr>
<td>cc65&lt;sup&gt;[28]&lt;/sup&gt;</td>
<td>$2^{23}$</td>
<td>65793 (101011&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>4282663 (415927&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>bits 22..8</td>
</tr>
<tr>
<td>cc65</td>
<td>$2^{32}$</td>
<td>16843009 (10101011&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>826366247 (31415927&lt;sub&gt;16&lt;/sub&gt;)</td>
<td>bits 31..16</td>
</tr>
</tbody>
</table>

Formerly common: RANDU<sup>[9]</sup>

<table>
<thead>
<tr>
<th>Source</th>
<th>modulus $m$</th>
<th>multiplier $a$</th>
<th>increment $c$</th>
<th>output bits of seed in $\operatorname{rand}()$ or $\operatorname{Random}(L)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Advanced Algorithms, A4M33PAL, ZS 20152016, FEL ČVUT
Many generators produce a sequence \( \{x_n\} \) defined by the general recurrence rule

\[
x_{n+1} = f(x_n) \quad n \geq 0.
\]

Therefore, if \( x_n = x_{n+k} \) for some \( k > 0 \), then also

\[
x_{n+1} = x_{n+k+1}, \quad x_{n+2} = x_{n+k+2}, \quad x_{n+3} = x_{n+k+3}, \ldots
\]

**Sequence period**

Subsequence of minimum possible length \( p > 0 \), \( \{x_n, x_{n+1}, x_{n+2}, \ldots x_{n+p-1}\} \) such that for any \( n \geq 0 \):

\[
x_n = x_{n+p}.
\]
Combined Linear Congruential Generator

Definition

Let there be \( r \) linear congruential generators defined by relations

\[
0 \leq y_{k,0} < M_k \\
y_{k,n+1} = (A_k y_{k,n} + C_k) \mod M_k, \quad n \geq 0, \\
1 \leq k \leq r.
\]

The combined linear congruential generator is a sequence \( \{x_n\} \) defined by

\[
x_n = (y_{1,n} - y_{2,n} + y_{3,n} - y_{4,n} + \ldots (-1)^{r-1} \cdot y_{r,n}) \mod (M_1 - 1), \quad n \geq 0.
\]

Fact

Maximum possible period length (not always attained!) is

\[
(M_1 - 1)(M_2 - 1) \ldots (M_r - 1) / 2^{r-1}.
\]

Example 7

\( r = 2, \quad 1 \leq y_{1,0} \leq 2147483562, \quad 1 \leq y_{2,0} \leq 2147483398 \)

\[
y_{1,n+1} = (40014y_{1,n} + 0) \mod 2147483563, \quad n \geq 0, \\
y_{2,n+1} = (40692y_{2,n} + 0) \mod 2147483399, \quad n \geq 0,
\]

\[
x_n = (y_{1,n} - y_{2,n}) \mod 2147483562, \quad n \geq 0.
\]

Period length is \( \frac{(M_1-1)(M_2-1)}{2} = 2305842648436451838 \).
Combined Linear Congruential Generator

Example 8 \( r = 3 \), \( y_{1,0} = y_{2,0} = y_{3,0} = 1 \),

\[
y_{1,n+1} = (9y_{1,n} + 11) \mod 16, \quad n \geq 0,
\]

\[
y_{2,n+1} = (7y_{2,n} + 5) \mod 18, \quad n \geq 0,
\]

\[
y_{3,n+1} = (4y_{3,n} + 8) \mod 27, \quad n \geq 0,
\]

\[
x_n = (y_{1,n} - y_{2,n} + y_{3,n}) \mod 15, \quad n \geq 0.
\]

\{x_n\} = 1, 4, 0, 2, 7, 12, 2, 2, 6, 6, 7, 7, 5, 2, 0, 9, 1, 1, 9, 11, 7, 9, 2, 8, 9, 12, 1, 1, 14, 2, 12, 9, 7, 4, 9, 8, 1, 6, 14, 5, 9, 0, 1, 4, 8, 8, 6, 9, 4, 4, 3, 11, 4, 3, 11, 14, 9, 12, 1, 7, 11, 11, 0, 0, 1, 1, 0, 11, 10, 3, 11, 11, 3, 6, 1, 4, 11, 2, 3, 6, 10, 10, 9, 11, 7, 3, 2, 14, 3, 3, 10, 1, 8, 14, 3, 9, 10, 13, 3, 2, 1, 3, 14, 14, 12, 6, 13, 13, 5, 8, 3, 6, 10, 1, 6, 5, 10, 9, 11, 11, 9, 6, 4, 13, 5, 5, 12, 0, 10, 13, 6, 11, 13, 0, 5, 5, 3, 6, 1, 13, 11, 8, 12, 12, 4, 10, 3, 8, 13, 3, 5, 8, 12, 12, 10, 13, 8, 8, 6, 0, 7, 7, 0, 2, 13, 0, 5, 11, 0, 0, 4, 4, 5, 5, 3, 0, 13, 7, 0, 14, 7, 9, 5, 8, 0, 6, 7, 10, 14, 14, 12, 0, 10, 7, 6, 2, 7, 6, 14, 5, 12, 3, 7, 13, 14, 2, 6, 6, 4, 7, 3, 2, 1, 9, 2, 2, 9, 12, 7, 10, 14, 5, 9, 9, 13, 13, 0, 14, 13, 9, 8, 2, 9, 9, 1, 4, 14, 2, 9, 0, 1, 4, 9, 8, 7, 9, 5, 2, 0, 12, 1, 1, 8, 14, 6, 12, 1, 7, 9, 11, 1, 0, 14, 2, 12, 12, 10, 4, 11, 11, 3, 6, 1, 4, 9, 14, 4, 3, 8, 8, 9, 9, 7, 4, 2, 11, 3, 3, 10, 13, 9, 11, 4, 9, 11, 14, 3, 3, 1, 4, 14, 11, 9, 6, 10, 10, 3, 8, 1, 6, 11, 2, 3, 6, 10, 10, 10, 8, 11, 6, 6, 4, 13, 6, 5, 13, 0, 11, 14, 3, 9, 13, 13, 2, 2, 3, 3, 1, 13, 12, 5, 13, 12, 5, 8, 3, 6, 13, 4, 5, 8, 12, 12, 10, 13, 9, 5, 4, 0, 5, 5, 12, 3, 10, 1, 5, 11, 12, 0, 4, 4, 3, 5, 1, 0, 14, 8, 0, 0, 7, 10, 5, 8, 12, 3, 7, 7, 12, 11, 13, 12, 11, 8, 6, 0, 7, 7, 14, 2, 12, 0, 7, 13, 0, 2, 7, 6, 5, 8, 3, 0, 13, 10, 14, 14, 14, 12, 4, 10, 0, 5, 7, 9, 14, 14, 12, 0, 10, 10, 8, 2, 9, 9, (sequence restarts:) 1, 4, 0, 2, 7, 12, 2, 2, 7, 7, 5, ...

Period length is \( 432 < 15 \cdot 17 \cdot 26 / 4 \).
Lehmer Generator

Lehmer generator produces sequence \( \{x_n\} \) defined by relations

\[
x_{n+1} = A x_n \mod M, \quad n \geq 0.
\]

Modulus \( M \), seed \( x_0 \), multiplier \( A \).

Example 9

\[
x_0 = 1,
\]

\[
x_{n+1} = 6x_n \mod 13.
\]

\( \{x_n\} = 1, 6, 10, 8, 9, 2, 12, 7, 3, 5, 4, 11, 1, 6, 10, 8, 9, 2, 12, \ldots \)

sequence period, length = 12

Example 10

\[
x_0 = 2,
\]

\[
x_{n+1} = 5x_n \mod 13.
\]

\( \{x_n\} = 2, 10, 11, 3, 2, 10, 11, 3, 2, 10, 11, 3, \ldots \)

sequence period, length = 4
Lehmer Generator

The sequence period length produced by a Lehmer generator is maximal and equal to $M - 1$ if $M$ is prime and $A$ is a primitive root of $(\mathbb{Z}/M\mathbb{Z})^*$. 

$x_{n+1} = Ax_n \mod M,$ \hspace{1cm} $n \geq 0.$

Fact

The sequence period length produced by a Lehmer generator is maximal and equal to $M - 1$ if $M$ is prime and $A$ is a primitive root of $(\mathbb{Z}/M\mathbb{Z})^*$.

Notation

Multiplicative group of integers modulo prime $p$: $(\mathbb{Z}/p\mathbb{Z})^*$

Primitive root

$G$ is a primitive root of $(\mathbb{Z}/p\mathbb{Z})^*$ if 

\[ \{G, G^2, G^3, ..., G^{p-1}\} = \{1, 2, 3, ..., p-1\} \]  (powers are taken modulo $p$).

Example 11

$p = 13, G = 2$ is a primitive root of $(\mathbb{Z}/13\mathbb{Z})^*$. 

\[ \{G, G^2, ..., G^{12}\} = \{2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7, 1\} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}. \]

$p = 13, G = 6$ is a primitive root of $(\mathbb{Z}/13\mathbb{Z})^*$. 

\[ \{G, G^2, ..., G^{12}\} = \{6, 10, 8, 9, 2, 12, 7, 3, 5, 4, 11, 1\} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}. \]

$p = 13, G = 5$ is not a primitive root of $(\mathbb{Z}/13\mathbb{Z})^*$. 

\[ \{G, G^2, ..., G^{12}\} = \{5, 12, 8, 1, 5, 12, 8, 1, 5, 12, 8, 1\} = \{1, 5, 8, 12\}. \]
Lehmer Generator

Finding group primitive roots

No elementary and effective method is known. Some cases has been studied in detail.

8th Mersenne prime \(M_{31} = 2^{31} - 1 = 2\,147\,483\,647\)

Fact \(G\) is a primitive root of \((\mathbb{Z}/M_{31}\mathbb{Z})^*\) iff 
\[
G \equiv 7^b \pmod{M_{31}}, \text{ where } b \text{ is coprime to } M_{31} - 1.
\]

\[
M_{31} - 1 = 2\,147\,483\,646 = 2 \cdot 3^2 \cdot 7 \cdot 11 \cdot 31 \cdot 151 \cdot 331
\]

Example 12

\(G = 7^5 = 16807\) is a primitive root of \((\mathbb{Z}/M_{31}\mathbb{Z})^*\) because 5 is coprime to \(M_{31} - 1\).

\(G = 7^{1116395447} \equiv 48271 \pmod{M_{31}}\) is a primitive root of \((\mathbb{Z}/M_{31}\mathbb{Z})^*\)
because 1116395447 is a prime and therefore coprime to \(M_{31} - 1\).

\(G = 7^{1058580763} \equiv 69621 \pmod{M_{31}}\) is a primitive root of \((\mathbb{Z}/M_{31}\mathbb{Z})^*\)
because 1058580763 = 19\cdot41\cdot61\cdot22277 and therefore coprime to \(M_{31} - 1\).
Blum Blum Shub generator produces sequence \( \{x_n\} \) defined by relations

\[
2 \leq x_0 < M, \quad x_0 \text{ coprime to } M.
\]
\[
x_{n+1} = x_n^2 \mod M
\]

Modulus \( M \), seed \( x_0 \).

Seed \( x_0 \) coprime to \( M \).
Modulus \( M \) is a product of two large distinct primes \( P \) and \( Q \).
\( P \mod 4 = Q \mod 4 = 3 \),
\( \gcd((P - 3)/2, (Q - 3)/2) \) is small.

**Example 13** \( x_0 = 4, \ M = 11 \cdot 47, \ \gcd(4, 22) = 2, \)
\[
x_{n+1} = x_n^2 \mod 517.
\]
\[
\]

sequence period, length = 44
Přesuňte 3 sirky tak, aby vlaštovka letěla na jih.

Přesuňte právě jednu z pěti modrých číslic, aby rovnost platila.

\[ 62 - 63 = 1 \]

Jaká dvojice písmen logicky patří na místo otazníků?

Vyřešte algebrogram.
Primes related notions

Prime counting function \( \pi(n) \)

Counts the number of prime numbers less than or equal to \( n \).

**Example 14**

\[ \pi(10) = 4. \text{ Primes less than or equal to 10: } 2, 3, 5, 7. \]
\[ \pi(37) = 12. \text{ Primes less than or equal to 37: } 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37. \]
\[ \pi(100) = 25. \text{ Primes less than or equal to 100: } 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97. \]

**Estimate**

\[
\frac{n}{\ln n} < \pi(n) < 1.25506 \frac{n}{\ln n} \quad \text{for } n > 16.
\]

**Example 15**

\[
\frac{100}{\ln 100} < \pi(100) < 1.25506 \frac{100}{\ln 100}
\]

\[
\frac{10^6}{\ln 10^6} < \pi(10^6) < 1.25506 \frac{10^6}{\ln 10^6}
\]

\[
21.715 < \pi(100) = 25 < 27.253
\]

\[
72382.4 < \pi(10^6) = 78498 < 90844.3
\]

**Limit behaviour**

Prime number theorem:

\[
\lim_{n \to \infty} \frac{\pi(n)}{\frac{n}{\ln n}} = 1
\]
Sieve of Eratosthenes

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
</tr>
<tr>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
</tr>
<tr>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
<td>80</td>
</tr>
<tr>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
</tr>
<tr>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
</tr>
</tbody>
</table>
### Sieve of Eratosthenes

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
</tr>
<tr>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
</tr>
<tr>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
<td>80</td>
</tr>
<tr>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
</tr>
<tr>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
</tr>
</tbody>
</table>
Sieve of Eratosthenes

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
</tr>
<tr>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
</tr>
<tr>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
<td>80</td>
</tr>
<tr>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
</tr>
<tr>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
</tr>
</tbody>
</table>
# Sieve of Eratosthenes

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
### Sieve of Eratosthenes

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
Sieve of Eratosthenes

Algorithm

EratosthenesSieve \((n)\)

Let \(A\) be an array of Boolean values, indexed by integers 2 to \(n\), initially all set to \text{true}

for \(i = 2\) to \(\sqrt{n}\)

    if \(A[i] = \text{true}\) then
        for \(j = i^2, i^2+i, i^2+2i, i^2+3i, \ldots\), not exceeding \(n\)
        
            \(A[j] := \text{false}\)
        
    end

output all \(i\) such that \(A[i]\) is \text{true}

end

Time complexity: \(O(n \log \log n)\).
Randomized primality tests

General scheme

\[ \text{n} \rightarrow \text{Test} \rightarrow \begin{cases} \text{Composite (definitely)} \\ \text{Prime (most likely)} \end{cases} \]

Fermat (little) theorem

If \( p \) is prime and \( 0 < a < p \), then \( a^{p-1} \equiv 1 \pmod{p} \).

Fermat primality test

\[
\text{FermatTest} (n, k) \\
\quad \text{for } i = 1 \text{ to } k \\
\quad \quad a = \text{random integer in } [2, n-2] \\
\quad \quad \text{if } a^{n-1} \not\equiv 1 \pmod{n} \text{ then return } \text{Composite} \\
\quad \text{end} \\
\text{return } \text{Prime} \\
\text{end}
\]

Flaw

There are infinitely many composite numbers for which the test always fails: Carmichael numbers: 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, ... (sequence A002997 in the OEIS)

Note

OEIS = The On-Line Encyclopedia of Integer Sequences, (https://oeis.org)
Randomized primality tests

Miller-Rabin primality test

**Fermat:** If \( p \) is prime and \( 0 < a < p \), then \( a^{p-1} \equiv 1 \pmod{p} \).

**Lemma:** If \( p \) is prime and \( x^2 \equiv 1 \pmod{p} \) then \( x \equiv 1 \pmod{p} \) or \( x \equiv -1 \pmod{p} \).

**Example:**
Is \( n = 15 \) prime?
Let \( a = 4 \).
Fermat test: \( 4^{15-1} \mod 15 = 1 \) ... OK.
Apply the lemma to \( 4^{14} \) --> If 15 is prime, then \( \sqrt{4^{14}} = 4^7 \mod 15 \in \{1, -1\} \). However, \( 4^7 \mod 15 = 4 \), hence 15 is a composite number.
Randomized primality tests

**Miller-Rabin primality test**

**Lemma:** If $p$ is prime and $x^2 \equiv 1 \pmod{p}$ then $x \equiv 1 \pmod{p}$ or $x \equiv -1 \pmod{p}$.

Let $n > 2$ be prime, $n-1 = 2^r \cdot d$ where $d$ is odd, $1 < a < n-1$.

Then either $a^d \equiv 1 \pmod{n}$ or $a^{2^s \cdot d} \equiv -1 \pmod{n}$ for some $0 \leq s \leq r - 1$.

MillerRabinTest $(n, k)$

compute $r, d$ such that $d$ is odd and $2^r \cdot d = n-1$

for $i = 1$ to $k$  // WitnessLoop

$a = \text{random integer in } [2, n-2]$

$x = a^d \mod n$

if $x = 1$ or $x = n-1$ then goto EndOfLoop

for $j = 1$ to $r-1$

$x = x^2 \mod n$

if $x = 1$ then return Composite

if $x = n-1$ then goto EndOfLoop

end

return Composite

EndOfLoop:

end

return Prime

end

**Examples:**

$n = 1105 = 2^4 \cdot 69+1$

$a = 389$

$x_0 = 1039$

$x_1 = 1041$

$x_2 = 781$

$x_3 = 1$ -> Composite

$n = 1105 = 2^4 \cdot 69+1$

$a = 390$

$x_0 = 539$

$x_1 = 1011$

$x_2 = 1101$

$x_3 = 16$ -> Composite

$n = 13 = 2^2 \cdot 3+1$

$a = 7$

$x_0 = 5$

$x_1 = 12 \equiv -1 \pmod{13}$

WitnessLoop passes
Randomized primality tests

Miller-Rabin primality test

- Time complexity: $O(k \log^3 n)$.
- If $n$ is composite then the test declares $n$ prime with a probability at most $4^{-k}$.
- A deterministic variant exists, however it relies on unproven generalized Riemann hypothesis.

AKS primality test

- First known deterministic polynomial-time primality test.
- Time complexity: $O(\log^6 n)$.
- The algorithm is of immense theoretical importance, but not used in practice.
Integer factorization

Difficulty of the problem

- No efficient algorithm is known.
- The presumed difficulty is at the heart of widely used algorithms in cryptography (RSA).

Pollard’s rho algorithm

- Effective for a composite number having a small prime factor.

PollardRho (n)

```plaintext
x = y = 2; d = 1
while d = 1
    x = g(x) mod n
g(x) .. a suitable polynomial function
    y = g(g(y)) mod n
For example, g(x) = x^2 - 1
d = gcd (|x-y|, n)
gcd .. the greatest common divisor
end
if d = n return Failure
else return d
end
```
Pollard’s rho algorithm – analysis

• Assume $n = pq$.
• Values of $x$ and $y$ form two sequences $\{x_k\}$ and $\{y_k\}$, respectively, where $y_k = x_{2k}$ for each $k$. Both sequences enter a cycle. This implies there is $t$ such that $y_t = x_t$.
• Sequences $\{x_k \mod p\}$ and $\{y_k \mod p\}$ typically enter a cycle of shorter length. If, for some $s < t$, $x_s \equiv y_s \pmod{p}$, then $p$ divides $|x_s - y_s|$ and the algorithm halts.

• The expected number of iterations is $O(\sqrt{p})=O(n^{1/4})$.

References


OEIS, The On-Line Encyclopedia of Integer Sequences (https://oeis.org)
