**Inceput**

**Operatia de exponentiere modulară**

**Algoritmul:**

z=1;

**for** i =s-1 **downto** 0 **do**

z= z2 **mod** n

**if** ci =1 **then** z=z\*x **mod** n

**Exemplul practic:**

7181^2953mod 9286= 9247

**c** = 2953

**x** = 7181

**n** = 9286

(2953)10 = (101110001001)2 = Ci

|  |  |  |
| --- | --- | --- |
| i | ci | Z=1 |
| 11 | 1 | **z= z2 mod n** = 12 mod 9286=1 \* 7181 mod 9286 = **7181** |
| 10 | 0 | 71812 mod 9286 = **1603** |
| 9 | 1 | 16032 mod 9286 = 6673 \* 7181 mod 9286 = **3053** |
| 8 | 1 | 30532 mod 9286 = 6951 \* 7181 mod 9286 = **2881** |
| 7 | 1 | 28812 mod 9286 = 7763 \* 7181 mod 9286 = **2245** |
| 6 | 0 | 22452 mod 9286 = **7013** |
| 5 | 0 | 70132 mod 9286 = **3513** |
| 4 | 0 | 35132 mod 9286 = **75** |
| 3 | 1 | 752 mod 9286 = 5625 \* 7181 mod 9286 = **8311** |
| 2 | 0 | 83112 mod 9286 = **3453** |
| 1 | 0 | 34532 mod 9286 = **9271** |
| 0 | 1 | 92712 mod 9286 = 225 \* 7181 mod 9286 = **9247** |

**Realizarea in limbajul java ase vedea Anexa 3**

Codul sursa pentru operatia de exponentiere modulara:

import java.util.Scanner;

public class BigPow {

public static void main(String[] args) {

System.out.println("Grad mare cu modulo");

System.out.println("a ^ k mod n (unde k - numar mare)");

System.out.println("Introduce a k n: ");

long a = readLong();

long k = readLong();

long n = readLong();

System.out.println("Resultat: " + pow\_mod(a, k, n));

}

/\*\* a ^ k mod n \*/

public static long pow\_mod(long a, long k, long n) {

long b = 1;

while (k > 0) {

if (k % 2 == 0) {

k /= 2;

a = (a \* a) % n;

} else {

k--;

b = (b \* a) % n;

}

}

return b;

}

public static long readLong() {

return new Scanner(System.in).nextLong();

}

}

**Exemplu:**

**Exponent mare modulo**

a ^ k mod n (unde k - numar mare)

Introduce a k n:

15

4947

31

Resultat: 23

**Sfirsit**

**Inceput**

**Algoritmul extins Euclid.**

**Algoritmul utilizat pentru calculul inversei unui numar mare modulo (b^(-1)mod n)**

**Algoritmul extins Euclid.**

1. n0 = n; b0 = b; t0 = 0; t = 1;

2.q = n0 / b0; r = n0 – q\*b0;

**3.while** r > 0 **do**

3.1.temp = t0 – q\*t;

3.2.**if** temp >= 0 **then** temp = temp **mod** n

**else** temp = n – ((- temp) **mod** n)

3.3n0 = b0; b0 = r; t0 = t; t= temp;

3.4.q = [n0/ b0]; r = n0 – q\*b0;

**4.if** b0 != 1 **then** b nu are inversă **mod** n

**else** b-1 **mod** n = t;

**Exemplul practic nr. 1:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | n0 | b0 | q | r |  | t0 | t | temp |
| 1 | 578 | 1282 | 0 | 578 |  | 0 | 1 | 0 |
| 2 | 1282 | 578 | 2 | 126 |  | 1 | 0 | 1 |
| 3 | 578 | 126 | 4 | 74 |  | 0 | 1 | 574 |
| 4 | 126 | 74 | 1 | 52 |  | 1 | 574 | 5 |
| 5 | 74 | 52 | 1 | 22 |  | 574 | 5 | 569 |
| 6 | 52 | 22 | 2 | 8 |  | 5 | 569 | 23 |
| 7 | 22 | 8 | 2 | 6 |  | 569 | 23 | 523 |
| 8 | 8 | 6 | 1 | 2 |  | 23 | 523 | 78 |
| 9 | 6 | **2** | 3 | 0 |  | 523 | 78 |  |

**Calcule matematice pentru fiecare rand din tabela:**

**1.** n0 = 578; b0 = 1282; t0 = 0; t = 1; q = 578 / 1282 = 0; r = n0 – q\*b0 = 578; temp = t0 – q\*t = 0 - 0\*1 = 0;

temp = temp mod n = 0 mod 578=0;

**2.** n0 = 1282; b0 = 578; t0 = 1; t= 0; q = 1282/ 578 = 2; r = 1282– 2\*578 = 126; temp = 1- 2\*0 = 1;

temp = 1 mod 578 = 1;

**3.** n0 = 578; b0 = 126; t0 = 0; t= 1; q = 578/ 126= 4; r = 578– 4\*126 = 74; temp = 0 – 4\*1 = - 4 ;

temp = n – ((- temp) mod n) = 578 – ((-(-4)) mod 578)= 578 – (4 mod 578 )= 574;

**4.** n0 = 126; b0 = 74; t0 = 1; t= 574; q = 126/ 74 = 1; r = 126– 1\*74 = 52; temp = 1 – 1\*574 = - 573; temp= 578 – ((-(-573)) mod 578) = 578 – ( 573 mod 578) = 5;

**5.** n0 = 74; b0 = 52; t0 = 574; t= 5; q = 74/ 52 = 1; r = 74 – 1\*52 = 22; temp = 574 – 1\*5 = 569;

temp = 569 mod 578 = 569;

**6.** n0 = 52; b0 = 22; t0 = 5; t= 569; q = 52/ 22 = 2; r = 52 – 2\*22 = 8; temp = 5 – 2\*569 = - 1133;

temp= 578 – ((-(- 1133)) mod 578) = 578 – (1133 mod 578) = 23;

**7.** n0 = 22; b0 = 8; t0 = 569; t= 23; q = 22/ 8 = 2; r = 22– 2\*8 = 6; temp = 569 – 2\*23 = 523;

temp = 523 mod 578 = 523;

**8.** n0 = 8; b0 = 6; t0 = 23; t= 523; q = 8/ 6 = 1; r = 8 – 1\*6 = 2; temp = 23 – 1\*523 = -500;

temp = 578 – ((-(-500) )mod 578) = 578 – (500 mod 578) = 578 – 500 = 78;

**9.** n0 =6; b0 = 2; t0 = 523; t= 78; q = 6/ 2 = 3; r = 6 – 3\*2 = 0;

Răspuns: **b nu are inversă, b0 != 1 , b0 = 2**

**Exemplul practic nr. 2:**

Realizarea numerica

1777-1 mod 2699 = **1853**

**b** = 1777

**n** = 2699

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | n0 | b0 | q | r | t0 | t | temp |
| 1 | 2699 | 1777 | 1 | 922 | 0 | 1 | 2698 |
| 2 | 5177 | 922 | 2 | 855 | 1 | 2698 | 2 |
| 3 | 922 | 855 | 1 | 67 | 2698 | 2 | 2696 |
| 4 | 855 | 67 | 12 | 51 | 2 | 2696 | 38 |
| 5 | 67 | 51 | 1 | 16 | 2696 | 38 | 2658 |
| 6 | 51 | 16 | 3 | 3 | 38 | 2658 | 161 |
| 7 | 16 | 3 | 5 | 1 | 2658 | 161 | **1853** |
| 8 | 3 | 1 | 3 | 0 | 161 | 1853 |  |

**Calcule matematice:**

**1.** n0 = 2699; b0 = 1777; t0 = 0; t = 1; q = [2699/ 1777 ]= 1; r = 2699 – 1\*1777 = 922; temp = 0 – 1\*1 = - 1;

temp = 2699 – ((-(- 1)) **mod** 2699) = 2699 – 1 = 2698 mod 2699=2698

**2.** n0 = 2699; b0 = 922; t0 = 1; t= 2698; q = 1777/ 922 = 2; r = 1777– 1\*922 = 855; temp = 1 – 1\*2698 = -2697;

temp = 2699 – ((-(- 2697)) **mod** 2699) = 2699 – (2697 mod 2699) = 2;

**3**. n0 = 922; b0 = 855; t0 = 2698; t= 2; q = 922/ 855 = 1; r = 922 – 1\*855 = 67; temp = 2698– 1\*2 = 2696;

4. n0 = 855; b0 = 67; t0 = 2; t=2696; q = 855/ 67 =12 ; r = 855 – 12\*67 = 51; temp = 2 – 12\*2696 = -32350; temp = 2699 – ((-(-32350)) **mod** 2699) = 2699 – (32350 mod 2699) = 38;

5. n0 = 67; b0 = 51; t0 = 2696; t= 38; q = 67/ 51 = 1; r = 67 – 1\*51 = 16; temp = 2696– 1\*38 = 2658;

6. n0 = 51; b0 = 16; t0 = 38; t= 2658; q = 51/ 16 = 3; r = 51 – 3\*16 = 3; temp = 38 – 3\*2658= - 7936;

temp = 2696 – ((-(-7936)) **mod** 2699) = 2699– (7936 mod 2699) = 161;

7. n0 = 16; b0 = 3; t0 = 2658; t=161; q = 16/ 3 = 5; r = 16 – 5\*3 = 1; temp = 2658 – 5\*161= 1853;

8. n0 = 3; b0 = 1; t0 = 161; t=1853; q = 3/1 = 3; r = 3 – 3\*1 = 0;

Răspuns: b-1 mod n = t ; 1777-1 mod 2699= 1853

**Realizarea in llimbajul java**

**Calculul inversei unui număr modulo**

Source:

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class AvMinus1ModP {

public static void main(String[] args) {

int b = 0, n = 0, n0 = 0, b0 = 0, q = 0, r = 1, t0 = 0, t = 0, temp = 0;

System.out.println(" b^-1 mod n ");

System.out.print("Introduce b (-1): ");

BufferedReader b1 = new BufferedReader(new InputStreamReader(System.in));

String inp = "";

try {

inp = b1.readLine();

} catch (IOException e) {

}

if (inp.equals("")) {

System.out.println("Nu era introdus nimic.");

System.exit(0);

}

b = Integer.parseInt(inp);

System.out.print("Introduce n: ");

try {

inp = b1.readLine();

} catch (IOException e) {

}

if (inp.equals("")) {

System.out.println("Nu era introdus nimic.");

System.exit(0);

}

n = Integer.parseInt(inp);

if (b <= 1 || n <= 0) {

System.out.println("b si n trebuie sa fie mai mari decit 0.");

System.exit(0);

}

// ################################################

System.out.println("b = " + Integer.toString(b) + ", n = "

+ Integer.toString(n) + "\n");

n0 = n;

b0 = b;

t0 = 0;

t = 1;

q = n0 / b0;

r = n0 - (q \* b0);

System.out.println("n0\tb0\tq\tr\tt0\tt\ttemp\n");

while (r > 0) {

temp = t0 - q \* t;

if (temp >= 0)

temp = temp % n;

else

temp = n - ((-temp) % n);

System.out.println(n0 + "\t" + b0 + "\t" + q + "\t" + r + "\t" + t0

+ "\t" + t + "\t" + temp);

n0 = b0;

b0 = r;

t0 = t;

t = temp;

q = n0 / b0;

r = n0 - (q \* b0);

}

System.out.println(n0 + "\t" + b0 + "\t" + q + "\t" + r + "\t" + t0

+ "\t" + t);

if (b0 != 1)

System.out.println("b nu are mod n");

else

System.out.println("Raspuns: " + t);

}

}

**Example:**

**b^-1 mod n**

**Introduce b (-1): 7**

**Introduce n: 11**

**b = 7, n = 11**

**n0 b0 q r t0 t temp**

**11 7 1 4 0 1 10**

**7 4 1 3 1 10 2**

**4 3 1 1 10 2 8**

**3 1 3 0 2 8**

**Raspuns: 8**

**Sfarsit**

**Grad mare modulo, exponentierea modulară**

Z=56171324 mod 9627

Transformăm exponentul în sistemul binar:

(1324)10=(10100101100)2, x=5617, n=9627

|  |  |  |
| --- | --- | --- |
| Nr | ci | Z=1 |
| 10 | 1 | Z=12 mod 9627= 1\*5617 mod 9627= 5617 |
| 9 | 0 | Z= 56172 mod 9627= 3010 |
| 8 | 1 | Z=30102 mod 9627= 9060100\*5617 mod 9627=6982 |
| 7 | 0 | Z=69822 mod 9627= 6823 |
| 6 | 0 | Z=68232 mod 9627= 6784 |
| 5 | 1 | Z=67842 mod 9627= 5596\*5617 mod 9627=577 |
| 4 | 0 | Z=5772 mod 9627= 5611 |
| 3 | 1 | Z=56112 mod 9627= 3031\*5617 mod 9627= 4591 |
| 2 | 1 | Z=45912 mod 9627= 3778\*5617 mod 9627= 3118 |
| 1 | 0 | Z=31182 mod 9627= 8281 |
| 0 | 0 | Z=82812 mod 9627= 1840 |

Răspuns: Z=56171324 mod 9627=1840

Codul sursa pentru operatia de exponentiere modulara:

import java.util.Scanner;

public class BigPow {

public static void main(String[] args) {

System.out.println("Grad mare cu modulo");

System.out.println("a ^ k mod n (unde k - numar mare)");

System.out.println("Introduce a k n: ");

long a = readLong();

long k = readLong();

long n = readLong();

System.out.println("Resultat: " + pow\_mod(a, k, n));

}

/\*\* a ^ k mod n \*/

public static long pow\_mod(long a, long k, long n) {

long b = 1;

while (k > 0) {

if (k % 2 == 0) {

k /= 2;

a = (a \* a) % n;

} else {

k--;

b = (b \* a) % n;

}

}

return b;

}

public static long readLong() {

return new Scanner(System.in).nextLong();

}

}

**Exemplu:**

**Exponent mare modulo**

a ^ k mod n (unde k - numar mare)

Introduce a k n:

15

4947

31

Resultat: 23

**Calculul inversei unui număr modulo**

Source:

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class AvMinus1ModP {

public static void main(String[] args) {

int b = 0, n = 0, n0 = 0, b0 = 0, q = 0, r = 1, t0 = 0, t = 0, temp = 0;

System.out.println(" b^-1 mod n ");

System.out.print("Introduce b (-1): ");

BufferedReader b1 = new BufferedReader(new InputStreamReader(System.in));

String inp = "";

try {

inp = b1.readLine();

} catch (IOException e) {

}

if (inp.equals("")) {

System.out.println("Nu era introdus nimic.");

System.exit(0);

}

b = Integer.parseInt(inp);

System.out.print("Introduce n: ");

try {

inp = b1.readLine();

} catch (IOException e) {

}

if (inp.equals("")) {

System.out.println("Nu era introdus nimic.");

System.exit(0);

}

n = Integer.parseInt(inp);

if (b <= 1 || n <= 0) {

System.out.println("b si n trebuie sa fie mai mari decit 0.");

System.exit(0);

}

// ################################################

System.out.println("b = " + Integer.toString(b) + ", n = "

+ Integer.toString(n) + "\n");

n0 = n;

b0 = b;

t0 = 0;

t = 1;

q = n0 / b0;

r = n0 - (q \* b0);

System.out.println("n0\tb0\tq\tr\tt0\tt\ttemp\n");

while (r > 0) {

temp = t0 - q \* t;

if (temp >= 0)

temp = temp % n;

else

temp = n - ((-temp) % n);

System.out.println(n0 + "\t" + b0 + "\t" + q + "\t" + r + "\t" + t0

+ "\t" + t + "\t" + temp);

n0 = b0;

b0 = r;

t0 = t;

t = temp;

q = n0 / b0;

r = n0 - (q \* b0);

}

System.out.println(n0 + "\t" + b0 + "\t" + q + "\t" + r + "\t" + t0

+ "\t" + t);

if (b0 != 1)

System.out.println("b nu are mod n");

else

System.out.println("Raspuns: " + t);

}

}

**Example:**

**b^-1 mod n**

**Introduce b (-1): 7**

**Introduce n: 11**

**b = 7, n = 11**

**n0 b0 q r t0 t temp**

**11 7 1 4 0 1 10**

**7 4 1 3 1 10 2**

**4 3 1 1 10 2 8**

**3 1 3 0 2 8**

**Raspuns: 8**

**Calculul logaritmului discret modulo**

**Logarifmul discret**

Source:

import java.util.Scanner;

public class ModulDeGradMare {

private static final int LIMIT = 32000;

public static void main(String[] args) {

int a=0;

int b=0;

int p=0;

try {

System.out.println("Logarifmul discret");

System.out.println(" a ^ x === b (mod p)");

System.out.println("Exemplu: 31 ^ 5 === 3 (mod 29)");

System.out.println("Introduce a : ");

a = (new Scanner(System.in).nextInt());

System.out.println("Introduce b : ");

b = (new Scanner(System.in).nextInt());

System.out.println("Introduce p : ");

p = (new Scanner(System.in).nextInt());

} catch (Exception e) {

System.out.println("Introduse datele incorecte. Numai cifrele pot fi introduse.");

System.exit(1);

}

if (a < 1 || b < 1 || p < 1) {

System.out.println("Datele introduce sunt incorecte. Numai cifrele positive pot fi introduse.");

System.exit(1);

}

int raspuns = aInXModPEqB(a, p, b, LIMIT);

if (raspuns >= 0) {

System.out.println("Raspuns : " + raspuns + "\r\n");

System.out.println(" " + a + " ^ " + raspuns + " === " + b

+ " (mod " + p + ")");

} else {

System.out.println("Raspuns nu a fost gasit.");

}

}

/\*\* a ^ x === b (mod p) \*/

public static int aInXModPEqB(int a, int p, int b, int limit) {

for (int i = 1; i < limit; i++) {

if (pow\_mod(a, i, p) == b)

return i;

}

return -1;

}

/\*\* a ^ k mod n \*/

public static long pow\_mod(long a, long k, long n) {

long b = 1;

while (k > 0) {

if (k % 2 == 0) {

k /= 2;

a = (a \* a) % n;

} else {

k--;

b = (b \* a) % n;

}

}

return b;

}}

**Example:**

Logarifmul discret

a ^ x === b (mod p)

Exemplu: 31 ^ 5 === 3 (mod 29)

Introduce a :

31

Introduce b :

3

Introduce p :

29

Raspuns : 5

31 ^ 5 === 3 (mod 29)