LABORATOR NR. 3

18. Maşini rotative

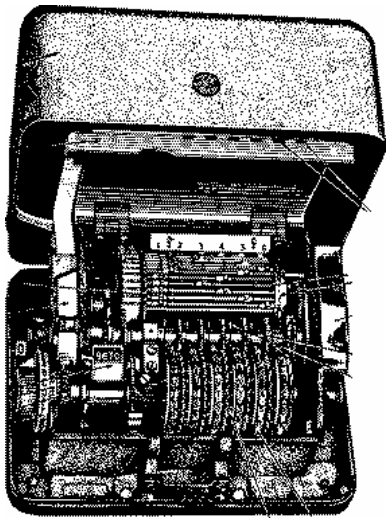


Figura 11. Maşina rotativă Enigma

După cum s-a precizat anterior o criptare multiplă face criptanaliza mult mai dificilă. Acest lucru este valabil atât pentru algoritmii de substituție cât şi pentru cei de transpoziție. Înaintea introducerii algoritmului DES, cea mai importantă aplicație care folosea tehnica criptării multiple era aceea a maşinilor rotative.

Una dintre acestea a fost şi maşina rotativă ENIGMA folosită foarte des în cel de-al Doilea Război Mondial şi prezentată în figura 11.

Maşina constă dintr-un set de cilindri rotativi care se pot roti independent, dotați cu conexiuni electrice. Fiecare cilindru are 26 de pini de intrare şi 26 pini de ieşire. Pinii de la intrare sunt conectați arbitrar cu cei de la ieşire, fiecare pin de la intrare având un unic corespondent la ieşire. Pentru simplificarea schemei sunt prezentate doar trei conexiuni.

Dacă asociem fiecare intrare şi ieşire cu o literă din alfabet, un singur cilindru realizează o substituție monoalfabetică. Dacă operatorul apasă tasta A, un semnal electric este aplicat la primul pin de la intrare al primului cilindru care este conectat intern la pinul 24 de la ieşirea primului cilindru.

Să considerăm un singur cilindru. După apăsarea unei taste cilindrul se roteşte cu o poziție, conexiunile interne fiind şi ele deplasate cu o poziție. Astfel se defineşte o substituție monoalfabetică diferită. După 26 de taste apăsate, cilindrul ajunge în poziția inițială. Rezultatul este o substituție polialfabetică cu perioada de 26 caractere.

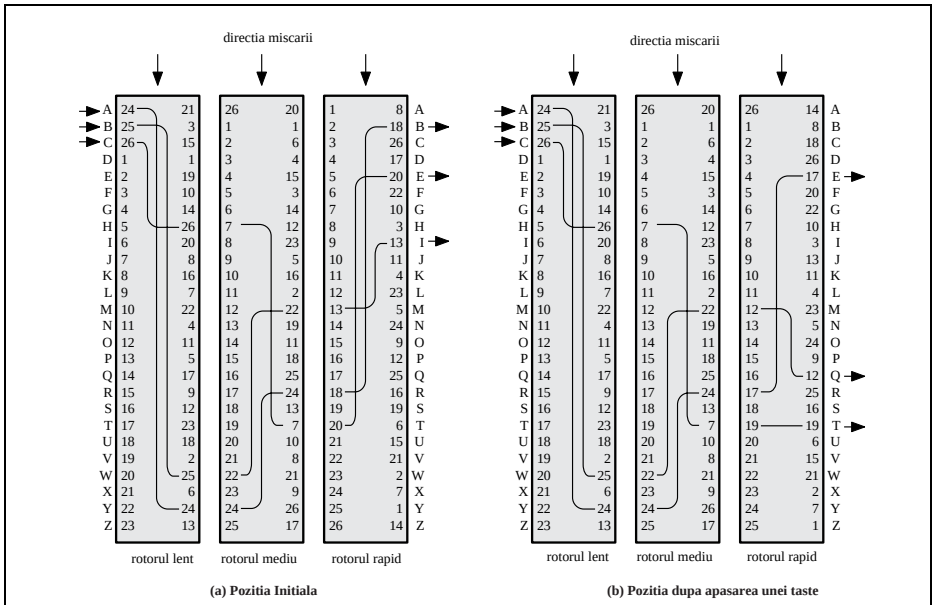


Figura 12. Mecanismul de funcţionare a maşinilor rotative

Un sistem cu un singur cilindru este banal nefiind dificil de decriptat. Puterea maşinilor rotative constă în folosirea mai multor cilindri, în care pinii de ieşire a unui cilindru sunt conectaţi la pinii de intrare a următorului cilindru, după cum s-a putut observa şi din figura 12. Partea din stânga a figurii arată cilindrii în poziţia iniţială, iar cea din dreapta după apăsarea unei taste.

Într-un sistem cu trei cilindri, cilindrul din dreapta (vezi figura 12) se roteşte o poziţie la fiecare tastă apăsată. Cilindrul din mijloc se roteşte o poziţie la fiecare rotaţie completă a cilindrului din dreapta lui. Acelaşi lucru se întâmplă şi cu cilindrul din stânga care se roteşte o poziţie la o rotaţie completă a cilindrului din mijloc. Rezultatul este un alfabet de substituţie cu 26 x 26 x 26 = 17576 substituţii diferite înainte ca acesta să se repete. Adăugând al patrulea şi al cincilea cilindru se poate lărgi şi mai mult plaja substituţiilor posibile. Pentru un sistem cu patru cilindri perioada se extinde la 456976 posibilităţi, iar la unul cu cinci cilindri la 11881376 posibilităţi.

Maşinile rotative au o importanţă deosebită deoarece au contribuit la dezvoltarea algoritmilor moderni de criptare şi în special a algoritmului DES (Data Encryption Standard) unul dintre cele mai răspândite şi mai des folosite metode de criptare din lume.

Criptarea cu cheie publică. Prezentare generală

În urma apariţiei în 1976 a articolului lui Diffie şi Hellman, *New Directions in Cryptography*, acest domeniu al criptării cu cheie publică s-a dezvoltat mult până în zilele noastre, foarte multe sisteme reale fiind implementate pe acest model de securitate. Este încă un domeniu de actualitate, continuându-se cercetările în laboratoare de specialitate cum ar fi RSA Laboratories şi Certicom.

Acest tip de criptare a apărut ca un efect al necesităţii comunicării de informaţie confidenţială, în lipsa unui canal sigur de comunicare, completând marele neajuns al criptării convenţionale: transmiterea cheii secrete. În cazul algoritmilor de criptare cu cheie secretă, indiferent de puterea acestor algoritmi, de viteza lor, transmiterea cheii secrete de la sursă către destinaţie devenea o mare problemă datorită posibilităţii de interceptare a acesteia de către o a treia parte.

O primă deosebire între cele două tipuri de criptare, cu cheie secretă şi cu cheie publică, este că cea cu cheie publică este asimetrică spre deosebire de cealaltă care este simetrică. După cum am arătat în paragraful anterior, o metodă de criptare simetrică constă într-un algoritm de criptare inversabil şi o cheie secretă, şi bineînțeles un algoritm de decriptare (inversul celui de criptare) şi o aceeaşi cheie secretă. Frumusețea şi ingeniozitatea metodei cu chei publice constă în faptul că mesajul este criptat cu ajutorul unei chei numită *publică*, iar decriptarea se face doar cu altă cheie numită *privată*. În zilele noastre există deja companii de încredere care generează aceste chei şi publică cheia *publică*, oferind astfel certificarea unui legitim schimb de mesaje între o anumită sursă şi destinație. Pentru a transmite un mesaj de la sursă către destinație, sursa va citi cheia publică a destinației prin intermediul respectivei companii de încredere, apoi va cripta mesajul cu respectiva cheie, cu ajutorul unui algoritm stabilit de comun acord şi public în acelaşi timp şi în cele din urmă va transmite mesajul către destinație prin intermediul unui canal de comunicație nu neapărat securizat. Mesajul criptat ajuns la destinație, urmează a fi decriptat cu ajutorul aceluiaşi algoritm cu care a fost criptat, dar folosind cheia privată a destinatarului. Gradul ridicat de securitate al algoritmilor cu chei publice se bazează pe faptul că, chiar dacă mesajul criptat este interceptat de către o a treia parte, acesteia îi va fi imposibil să îl decripteze în timp util şi cu un cost convenabil, fără cunoaşterea cheii private a destinatarului.

O importanță deosebită în alegerea algoritmului de criptare şi a dimensiunii cheilor private şi publice o au durata de valabilitate şi importanța mesajului transmis. Un algoritm de criptare cu cheie publică este cu atât mai sigur cu cât dimensiunea cheilor este mai mare, deci siguranța este direct proporțională cu dimensiunea cheilor. Pe de altă parte cu cât dimensiunea cheilor creşte, va creşte şi durata de execuție a algoritmului, deci va scădea viteza. Putem concluziona, că siguranța unui algoritm de criptare este direct proporțională cu dimensiunea cheilor de criptare/decriptare şi invers proporțională cu viteza de execuție. Din acest motiv, un prim pas, foarte important este alegerea dimensiunii cheilor. De exemplu dacă dorim transmiterea unor mesaje prin care se vor cumpăra acțiuni, care ne facilitează câştigul unei sume de 100000 de unități monetare, iar timpul limită pentru achiziționarea acțiunilor este de o zi, timpul de valabilitate al mesajului este de exact o zi, ceea ce înseamnă că dacă mesajul a fost interceptat de o a treia parte, aceasta va trebui să îl descifreze în maxim o zi, pentru că depăşind acest interval informația nu îi va mai fi utilă de loc. Dată fiind posibilitatea interceptării mesajului, cele două parți nu vor avea decât să urmărească două variabile: durata de valabilitate a mesajului şi costul unui atac în timp util. De exemplu, dacă dimensiunea cheilor ar fi de 30 de biți şi dacă un laborator specializat dotat cu 100 de calculatoare lucrând în paralel ar reuşi spargerea mesajului în 8 ore, practicând un tarif de 10000 unități monetare pe oră, ar fi suficient mărirea cu puțin al dimensiunii cheilor pentru ca atacul să nu-şi merite efortul în unitatea de timp.

În figura 13 este prezentat succint modelul criptării cu chei publice:

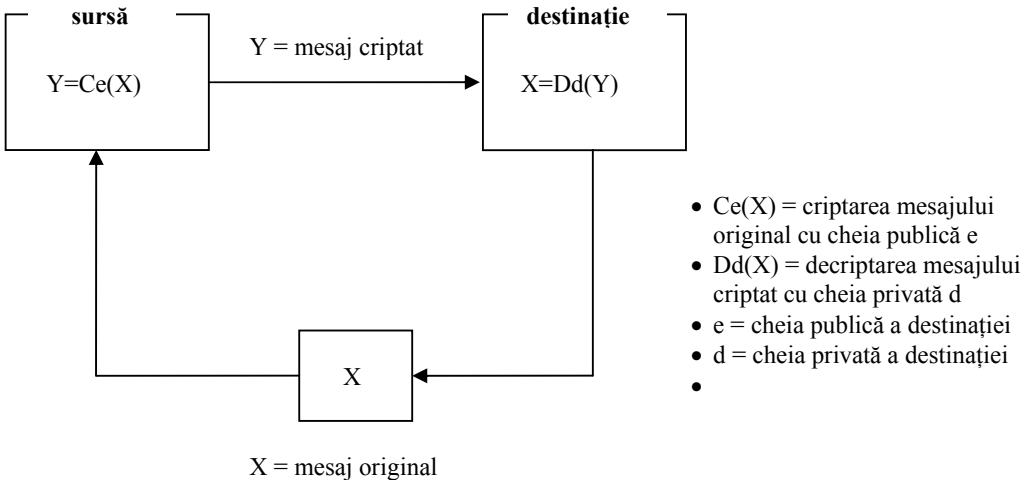


Fig. 13

Metoda RSA

Algoritmul RSA, numit după cei trei cercetători de la Massachussets Institute of Technology, Rivest, Shamir, Adleman, a fost publicat în anul 1978, fiind primul algoritm cu chei publice de o importanță deosebită.

Datorită problemei matematice care stă la baza criptosistemului RSA, această metodă de criptare este folosită în zilele noastre pe scară largă în plățile electronice, servere web, autentificări, semnături digitate, etc.

Deşi publicată cu peste douăzeci şi cinci de ani în urmă şi deşi s-a investit mult timp şi mulți bani în cercetarea securității acestei metode de criptare, încă nu s-a găsit o metodă generală şi polinomială de spargere a criptosistemelor RSA. De-a lungul timpului, pornind de la această metodă au apărut mai multe tehnici de criptare, dar nu toate s-au dovedit a fi la fel de puternice ca şi metoda RSA de bază.

Principiul de funcționare este identic cu cel propus de Diffie şi Hellman, schimbul de informație confidențială între două părți(să zicem A şi B) se face prin codificarea mesajului de către transmițător cu cheia publică a receptorului, acesta din urmă urmând să decodifice mesajul cu ajutorul cheii sale private. Ambele părți vor urma următorii paşi:

- se aleg aleator două numere prime *p* şi *q* de dimensiune *l/2.*
- se calculează *n=pq*
- se alege un număr *e* care să fie prim cu *(p-1)(q-1)*, adică *cmmdc(e, (p-1)(q-1))=1*
- se alege *d*, astfel încât $d = e^{-1} \bmod((p-1)(q-1))$

După respectarea acestor paşi, cheia publică va fi (e, n) şi cheia secretă (d,n), iar criptarea şi decriptarea vor fi doar nişte simple operații de exponențiere. RSA fiind un cod bloc, mesajul inițial va fi împărțit în bucăți care să nu depăşească însă *l-1* biți.

Presupunând că un bloc al mesajului inițial este *x*, criptarea acestui bloc se va face astfel:

$$y = x^e \bmod n,$$

iar decriptarea lui y:

$$x = y^d \bmod n.$$

Folosind aceste două relații pentru criptare şi decriptare şi considerând alegerea lui *e* şi a lui *d* precum s-a arătat mai sus, putem deduce matematic că mesajul codat de către sursă şi transmis către destinație va fi identic cu cel decriptat la destinație:

$$y = x^e \bmod n \qquad (1)$$
$$d = e^{-1} \bmod((p-1)(q-1)) \qquad (2)$$

Din (1) şi(2) retultă că:

$$y^d = (x^e \bmod n)^d = x^{ed} \bmod n = x \bmod n = x$$

Având la bază o problemă matematică grea, factorizarea întregilor mari, este esențială dimensiunilor numerelor prime *p* şi *q* alese. Cu cât aceste numere sunt mai mari cu atât criptosistemul este mai sigur, bineînțeles dacă se respectă anumite reguli şi nu se fac greşeli prin care problema matematică se reduce la cazuri particulare şi astfel criptosistemul este expus atacurilor**.**

RSA. Modelul criptosistemului simplu

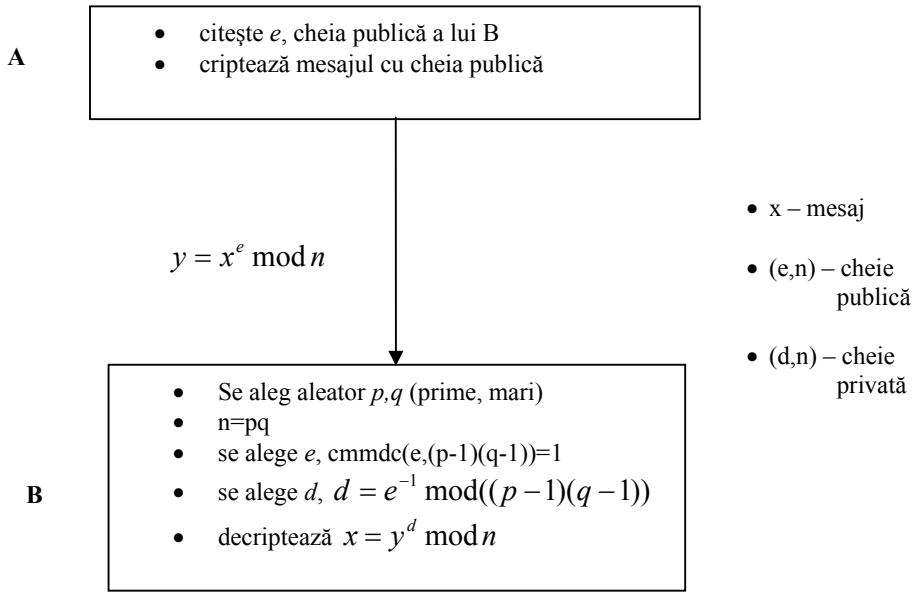


fig. 14

După cum se poate observa din figura 14, pentru a transmite un mesaj de la A către B, mai întâi B va genera aleator două numere prime mari (*p* şi *q*), apoi va calcula *n=pq*, distrugând cele două numere prime pentru a se evita interceptarea lor de către un eventual

adversar. Se alege un exponent de criptare *e*, cu proprietatea că cel mai mare divizor comun dintre el şi produsul (p-1)(q-1) este 1, apoi în funcție de acest *e*, se alege un exponent de decriptare *d*, cu proprietatea $d = e^{-1} \bmod((p-1)(q-1))$.

Mesajul inițial *x* este criptat mai apoi cu ajutorul exponentului de criptare *e*, de fapt cheia publică a lui B. Criptarea se face ridicând valoarea numerică a mesajului *x* la puterea *e*, totul modul *n.* RSA fiind un cod bloc, va necesita împărțirea mesajului inițial în blocuri de dimensiuni mai mici decât *n*, pentru a se putea realiza operațiile *modulo n*. Dimensiunile blocurilor pot fi determinate încă de la începutul algoritmului deoarece ce cunosc dimensiunile pe biți ale numerelor prime generate *p* şi *q*, să zicem *l/2*, deci dimensiunea lui *n=pq* este *l*. Cunoscând astfel lungimea pe biți a lui *n*, putem împărți mesajul inițial în blocuri de lungime *l-1*, fără a risca operații care să depăşească *modulo n*.

Datorită relației conform căreia a fost ales exponentul de decriptare *d*, $d = e^{-1} \bmod((p-1)(q-1))$, mesajul original va putea fi recuperat cu exactitate din mesajul codat prin ridicarea la puterea *d* a valorii numerice a mesajului codat.

Operațiile folosite de acest model simplu de criptare RSA sunt:

- Generarea de numere prime mari, aleator
- Testarea primalității
- Algoritm de calculare al celui mai mare divizor comun
- Ridicarea la putere modulo n

Calculele necesare criptării/decriptării nu sunt mari, dar nici foarte mici, trebuind ales cu foarte mare atenție raportul dintre viteza de calcul necesară şi gradul de securitate. Fiind un algoritm de criptare foarte larg răspândit, este implementat atât în sisteme în care timpul de execuție este critic(codul este optimizat la maxim şi cheile sunt alese de dimensiuni cât mai mici cu condiția ca gradul de securitate să fie suficient de ridicat), cât şi în sisteme în care securitatea este primordială şi dimensiunea cheilor este mare (1024 chiar 2048 de biți).
Metoda RSA se potriveşte perfect pe modelul criptării cu chei publice propuse de Diffie şi Hellman, transformând mesajul inițial *x* în $x^e \bmod n$, de fapt o funcție cu sens unic, având ca "portiță de scăpare" exponentul de decriptare *d*. Această funcție cu sens unic poate fi uşor calculată, dar invers, determinarea lui *x* cunoscând doar valoarea $x^e \bmod n$ fără exponentul *d*, este aproape imposibil de inversat în timp util cu excepția câtorva cazuri particulare care pot fi uşor evitate. Bineînțeles criptosistemul nu este perfect şi astfel, de-a lungul timpului cercetătorii au descoperit diferite posibilități de atac în anumite cazuri particulare. Dificultatea rezolvării problemei factorizării nu poate fi comparată cu dificultatea spargerii unui criptosistem RSA, iar spargerea unui astfel de criptosistem nu garantează găsirea unei soluții a problemei factorizării întregilor mari. Atacurile RSA pot fi realizate printr-un studiu amănunțit al cheilor, prin interceptarea mai multor mesaje transmise cu aceeaşi cheie, prin insuficiența lungimii cheilor şi multe alte metode care fac din RSA un caz restrâns de factorizare. În schimb, dacă problema matematică a factorizării întregilor mari va fi rezolvată într-un timp polinomial, securitatea criptosistemelor RSA va deveni nulă.

RSA. Exemplu

Pentru a exemplifica metoda RSA vom lua un exemplu pur experimental cu lungimi ale cheilor de 8 biți. Se vor parcurge următorii paşi:

1. se aleg două numere prime *p* şi *q*
2. se alege *e*, astfel încât *cmmdc(e,(p-1)(q-1))=1*
3. se calculează *d*, astfel încât (*de-1*) divide *(p-1)(q-1)*
4. se criptează *y=x^e mod n*
5. se decriptează *x=y^d mod n*

Rezolvare:

1. Alegem *p* şi *q*

p = 53
q = 57
n = p * q = 53 * 57 = 3021
(p-1) * (q-1) = 52 * 56 = 2912

2. Alegem *e*

e = 3
cmmdc(3,2912)=1 - corect

3. Calculăm *d*

(*de-1*) divide *(p-1)(q-1)* ➔ *d=(x*(p-1)(q-1)+1)/e* – număr întreg
se pot găsi cu uşurinţă diferite valori pentru *d*(971, 3883, 6795, 9707,...)

alegem d=6795

4. Alegem pentru criptare x=95

y=(x^e) mod n
y=(95^3) mod 3021=857375 mod 3021 = 2432

5. Decriptăm mesajul 2432 cu ajutorul cheii private *d*

x=(y^d) mod n
x=(2432^6795) mod 3021

$$6795_{(10)} = 1101010001011_{(2)} = 2^0 + 2^1 + 2^3 + 2^7 + 2^9 + 2^{11} + 2^{12}$$

$$2432^{6795} = 2432^{2^0+2^1+2^3+2^7+2^9+2^{11}+2^{12}}$$

se calculează uşor 2432^{2^i} mod 3021, i=0..12 :

$$2432^{2^0} \bmod 3021 = 2432$$

$$2432^{2^1} \bmod 3021 = 2527$$

$$2432^{2^3} \bmod 3021 = 1159$$

$$2432^{2^7} \bmod 3021 = 2413$$

$$2432^{2^9} \bmod 3021 = 2983$$

$$2432^{2^{11}} \bmod 3021 = 646$$

$$2432^{2^{12}} \bmod 3021 = 418$$

➔ y = 2432*2527*1159*2413*2983*646*418 mod 3021 =
1384431982242573857331 2 mod 3021 = 95

Testarea primalității

Problema : Se dă un număr natural N, să se testeze dacă este prim sau nu.

Metode deterministe

Varianta 1

Se testează dacă N se imparte exact la cel putzin unul dintre numerele: 2,3,4,...N-1. Dacă se găseşte cel puțin un divizor, atunci N nu este prim, altfel N este prim

```
int prim(){
 for (i=2;i<=N-1;i++)
  if (N % i == 0) return 0;
 return 1;
}
```

Varianta 2

Se testează dacă N este impar şi dacă este, atunci testează dacă se împarte exact la cel putzin unul dintre numerele impare: 3,5,...N-1. Dacă se găseşte cel puțin un divizor, atunci N nu este prim, altfel N este prim.

```
int prim(){
 if (N % 2 == 0) return 0;
 for (i=3;i<=N-1;i+=2)
  if (N % i == 0) return 0;
 return 1;
}
```

Varianta 3

Se testează dacă N este impar şi dacă este, atunci testează dacă se împarte exact la cel putzin unul dintre numerele impare: 3,5,... SQRT(N). Dacă se găseşte cel puțin un divizor atunci N nu este prim, altfel N este prim. Observăm că se reduce considerabil numărul de potențiali divizori ai lui N

```
int prim(){
 if (N % 2 == 0) return 0;
 for (i=3;i<=SQRT(N);i+=2)
  if (N % i == 0) return 0;
 return 1;
}
```

Varianta 4

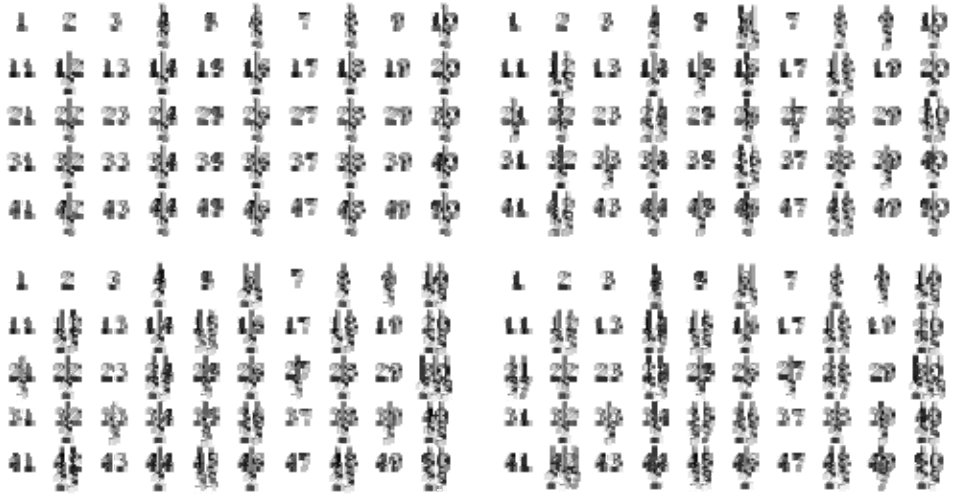
Plecând de la obervația că orice număr natural poate fi scris după formula (6*k+i), $i \in \{-1,0,1,2,3,4\}$, eliminând combinațiile divizibile cu 2 şi 3 obținem că numerele prime sunt de forma (6*k ±1). Testarea primalității este asemănătoare variantelor anterioare, diferența fiind timpul redus de căutare în spațiul soluțiilor

```
int prim(){
 if (N % 3 == 0) return 0;
 for (i=1;6*i+1<=N;i++)
  if (N % (6*i-1)==0 || N % (6*i+1)==0 ) return 0;
 if ((6*i-1<=N) && (N % (6*i-1)==0)) return 0;
 return 1;
}
```

O altă optimizare are fi generarea tuturor numerelor prime în prealabil evitând astfel împărțirii la divizori neprimi ca şi în variantele prezentate mai sus. O metodă simplă de genereare a numerelor prime mai mici decât un anumit număr dat se face cu metoda lui Eratostene. Conform metodei mai sus amintită plecând de la un vector cu toate numerele naturale între 2 şi un număr dat M, la fiecare pas al metodei se vor „taia" din vector toate numerele care se divid cu un anume număr prim:

- pasul 1 – se alege primul număr „netăiat"
- pasul 2 – se „taie" taie toate numerele care se divit cu numărul ales la pasul 1
- se repetă pasul 1 până când s-au „tăiat" toate numerele până la SQRT(M).

Exemplu pentru M=50,



Metode probabilistice

Aceste metode vor afirma cu un anumit grad de precizie dacă un număr este prim. Folosind aceste metode este posibil ca un număr compus să fie catalogat ca prim, dar dacă un număr a fost detectat ca neprim, atunci cu siguranță acel număr nu este prim.

Metoda Fermat

Conform teoremei lui Fermat, un număr N este prim dacă pentru orice a, $1 <= a < N$, are loc egalitatea $a^{N-1} = 1 (\mathrm{mod}\, N)$. Cu alte cuvinte, dacă găsim cel puțin un număr a, cu proprietatea $1 <= a < N$, astfel încât relația de mai sus să nu fie satisfăcută

($a^{N-1} \neq 1(\mathrm{mod}\, N)$), atunci putem spune ca N nu este număr prim. Pentru valori mari ale lui N, testarea relației din teoremă este foarte costisitoare ca timp. Din acest motiv, se vor face teste pentru anumite valori ale lui a, $1 <= a < N$, rezultatul fiind probabilistic. Ce este cert totusi este că dacă s-a găsit cel puțin o valoare a lui a pentru care relația de mai sus nu este satisfăcută atunci se poate afirma că numărul N nu este prim. Acestă metodă este rar folosită pentru că există anumite numere naturale compuse care satisfac aşa numita proprietate Carmichael, pe care metoda Fermat le recunoaşte ca prime.

Metoda Miller-Rabin

Ca principiu această metodă seamănă cu metoda Fermat prin faptul că verifică dacă se îndeplinesc anumite relații legate de numărul testat N.

Se calculează d, astfel încât:

$$N - 1 = 2^S d$$

Spunem că numărul N este compus dacă găsim un număr a, care să satisfacă următoarele:

$$a^d \neq 1(\mathrm{mod}\, n) \text{ şi } a^{2^r d} \neq -1(\mathrm{mod}\, n)$$

Metoda Solovay-Strassen

Asemănătoare cu metodele de mai sus, verifică dacă se îndeplinesc anumite relații legate de numărul testat N.

Spunem ca N este prim dacă pentru orice a se verifică relația:

$a^{(N-1)/2} = \left(\frac{a}{N}\right)(\mathrm{mod}\, N)$, $\left(\frac{a}{N}\right)$ - simbol Legendre, a<=N-1

$$\left(\frac{a}{N}\right) = \begin{cases} 0, \text{p divide pe a} \\ 1, \text{a este de forma } a = k^2 (\mathrm{mod}\, N) \\ -1, \text{a nu este de forma } a = k^2 (\mathrm{mod}\, N) \end{cases}$$