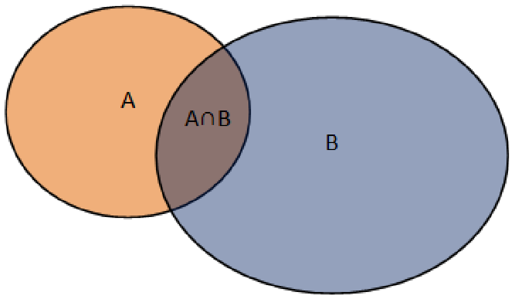
Formula uključivanja-isključivanja

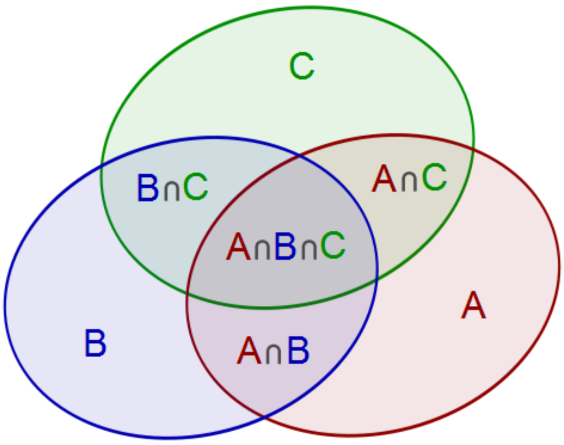
U informatičkim zadacima se zna pojaviti problem izračunavanja koliko elemenata ima unija više skupova, a da znamo izračunati koliko elemenata imaju njihovi presjeci. Pošto nas samo zanima koliko ima elemenata neki skup, onda ćemo pod skup S podrazumijevati broj elemenata skupa S.

Promotrimo kako izračunati uniju dva skupa.



Iz slike se vidi da je unija skupova A i B jednaka A + B - A∩B.

Promotrimo kako bismo izračunali uniju tri skupa.



Iz slike se vidi da je unija ovih skupova:
A + B + C +
– (A∩B + A∩C + B∩C) +
+A∩B∩C

Sada već naslućujemo da je unija četiri skupa:
A + B + C + D +
– (A∩B + A∩C + A∩D + B∩C + B∩D + C∩D) +
+ A∩B∩C + A∩B∩D + A∩C∩D + B∩C∩D +
– A∩B∩C∩D

Općeniti oblik formule uključivanja-isključivanja jest:

$$\left|\bigcup_{i=1}^{n} S_i\right| = \sum_{1\le i\le n} |S_i| - \sum_{1\le i<j\le n} |S_i \cap S_j| + \sum_{1\le i<j<k\le n} |S_i \cap S_j \cap S_k| + \ldots + (-1)^{n-1}\left|\bigcap_{1\le i\le n} S_i\right|$$

gdje je n broj skupova, a S_i su tih n skupova.

Navedenu formulu možemo jednostavnije iskazati riječima.
Rastavimo formulu na dva dijela:

a) Na sve načine odaberemo neparan broj skupova, te zbrojimo sume njihovih presjeka
b) Na sve načine odaberemo paran broj skupova, te zbrojimo sume njihovih presjeka

Te je broj članova unije N skupova razlika prve i druge sume.

Ovu formulu je moguće dokazati matematičkom indukcijom.

Primjer informatičkog zadatka

Koliko ima prirodnih brojeva manjih ili jednakih N, N <= 10^9 djeljivih s barem jednim brojem iz skupa S, koji ima najviše 18 elemenata?

Prirodnih brojeva <= N koji su djeljivi s nekim brojem X ima točno $\left\lfloor \frac{N}{X} \right\rfloor$.
Stoga možemo dodati u rješenje koliko je puta svaki broj iz skupa djeljiv sa N, ali ćemo na taj način dodati brojeve koje su djeljivi sa više brojeva iz skupa. Na primjer, ako su u skupu brojevi 4 i 6, te je N jednak 10. Brojevi djeljivi s 4 su 4, 8, **12**, 16, 20, **24**; a brojevi djeljivi s 6 su 6, **12**, **24**. Vidimo da smo brojeve 12 i 24 dodali dva puta. Znači da nakon što smo dodali brojeve djeljive s jednim brojem iz skupa, treba oduzeti brojeve koji su djeljivi sa dva broja iz skupa, pa opet dodati brojeve koji su djeljivi s tri broja iz skupa, …
Dobili smo znači princip uključivanja – isključivanja.

U računalu ćemo trenutačno stanje u predstaviti kao niz nula i jedinica. Ako imamo skup od četiri člana A, B, C i D, tada će stanje označavati da dodajemo ili oduzimamo presjek skupova na mjestima jedinica u stanju, npr. ako je stanje 1010, tada moramo oduzeti presjek skupova brojeva koji su djeljivi sa A i koji su djeljivi sa C, odnosno to znači da ćemo oduzeti $\left\lfloor \frac{N}{lcm\,(A,C)} \right\rfloor$. Prema broju jedinica u stanju određujemo hoćemo li rezultat oduzeti ili dodati. Kod ovog zadatka ako je paran broj jedinica onda oduzimamo vrijednost stanja, inače ga dodajemo.

Implementaciju u programskom jeziku C++

```
int sol = 0;
for( int stanje = 1; stanje < (1<<K); stanje++ ){ //prolazi po svim stanjima
  int predznak = -1;
  long long NZV = 1; //u ovoj varijabli pamtimo lcm
  for( int i = 0; i < K; i++){ //prolazi po svim članovima skupa
    if( stanje&(1<<i) ){ //ako je i-ti član stanja 1
      NZV = lcm( NZV, niz[i] ); //lcm računa najmanji zajednički višekratnik
      if( NZV > N ) break; //prekidamo računanje zbog owerflow-a
      predznak *= -1; //mijenjamo predznak
    }
  }
  sol += predznak * (N / NZV); //u C-u je dijeljenje cjelobrojno
}
```

Složenost ovakvih implementacija je O$(2^K * K)$, gdje je K veličina skupa S, ali baš ova implementacija ima složenost je O$(2^K * K * \ln N)$ jer se u unutarnjoj petlji računa najveći zajednički višekratnik brojeva manjih od N, čija je približna složenost $\ln N$.

Poboljšana implementacija

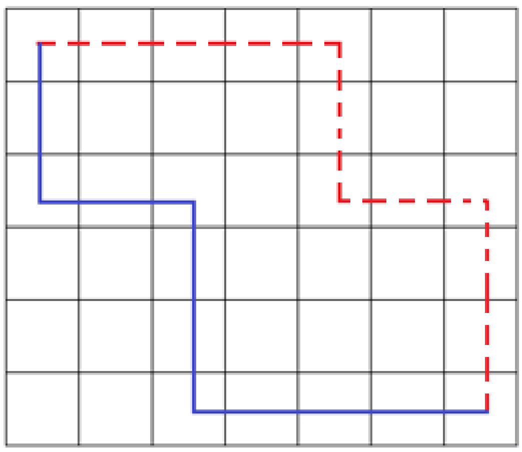
Složenost je moguće poboljšati eliminacijom unutarnje petlje, tako da ukupna složenost bude O($2^K * \ln N$). Takva implementacija zahtijeva poznavanje dinamičkog programiranja.
Da bismo izračunali vrijednost stanja 1010100, potrebno je znati vrijednost stanja 0010100. Stanje X računamo iz stanja u kojem je zadnja jedinica stanja X pretvorena u nulu.
Primijetimo da možemo stanja obrađivati od manjih prema većima jer se trenutačno stanje uvijek izračunava iz manjeg stanja.
Radi lakše implementacije, da bismo dobili prošlo stanje uvijek brišemo jedinicu sa najvećom vrijednosti.

```
DP[0] = 1;
Predznak[0] = -1;
int najveca = 0; //pozicija zadnje jedinice u stanju (1010) --> 3
for( int i = 1; i < (1<<K); i++ ){ //prolazi po stanjima od manjih prema većem
  if( ((1<<najveca)&i) == 0 ) najveca++;
  int prev = i - (1<<najveca); //prev je stanje iz kojeg računamo trenutacno
  long long V = lcm( DP[prev], niz[najveca] );
  Predznak[i] = -Predznak[prev];
  if( V > N ) DP[i] = N+1; //pazimo na owerflow
  else DP[i] = V;
  sol -= Predznak[i] * (N / DP[i]);
}
```

Još jedan informatički zadatak

Zamislimo pravokutnu matricu od R redaka i S stupaca ($R,S < 10^5$). Mi se nalazimo u gornjem lijevom kutu i cilj nam je doći do donjeg desnog kuta, te se možemo kretati samo dolje ili desno. U toj pravokutnoj matrici se nalazi K < 15 neprohodnih polja. Zanima nas na koliko se različitih načina može ostvariti taj put (od gornjeg lijevog kuta do donjeg desnog) modulo 1000007.

Kako izgledaju putovi ako su sva polja prohodna?

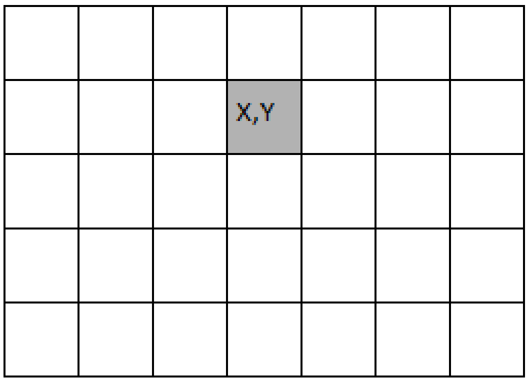


Ispišimo koje sve poteze radimo na svakom putu (R-prelazak na desno polje, D-prelazak na polje ispod).

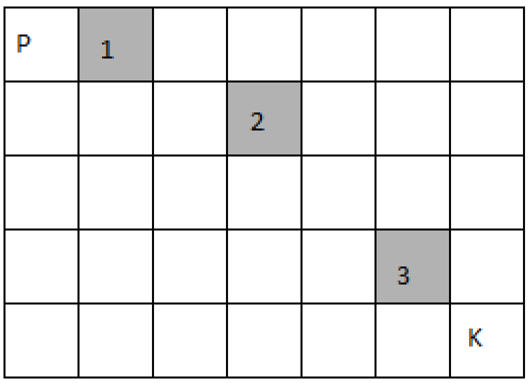
Iscrtkani put: RRRRDDRRDDD
Put s punom linijom: DDRRDDDRRRR
Uočavamo da je broj slova R u svakom putu jednak S-1, i broj slova D je R-1.
To znači da je broj različitih putova odgovor na pitanje na koliko načina možemo odabrati S-1 elemenata od S+R-2 elemenata. Takva funkcija se zove povrh i računa se kao: $\binom{S+R-2}{S-1} = \frac{(S+R-2)!}{(S-1)!(R-1)!}$. Od sada ćemo broj putova u pravokutniku veličine R*S označavati kao f(R,S).

Sada znamo izračunati koliko ima različitih putova u matrici veličine R*S, ako su svi jedinični kvadratići prohodni.

Koliko ima putova u matrici koji prolaze kroz točno određeni kvadratić?



Broj putova koji prolaze kroz sivi kvadratić je broj putova koji dolaze do sivog kvadratića puta broj putova koji iz sivog kvadratića odlaze u donji desni kut, odnosno f(X,Y) * f(R-X+1,S-Y+1).



Broj putova koji prolaze kroz sva tri kvadratića jednak je umnošku broju putova u pravokutnicima (P,1), (1,2), (2,3) i (3,K).
S tim da jedino treba paziti da je moguće obići sve kvadratiće jer da je kvadratić s brojem 2 lijevo od kvadratića s brojem 1, onda bi sveukupan broj putova koji prolaze kroz njih bio nula.

Sada već uočavamo rješenje pomoću pravila uključivanja-isključivanja.
Prvo treba dodati sve putove u cijeloj matrici, zatim treba oduzeti sve putove koji prolaze kroz jedan neprohodni kvadratić, zatim treba dodati one putove koji prolaze kroz dva neprohodna kvadratića…

Još postoji jedan problem koji je vezan uz dijeljenje brojeva modulo *p*, ali taj problem možemo pretvoriti u množenje formulom koja slijedi iz malog Fermatovog teorema:

$\frac{a}{b} = a * b^{p-2}\ (mod\ p), gdje\ je\ p\ prost\ broj\ i\ b\ nije\ djeljiv\ s\ p$

Znači da dijeljenje možemo riješiti u složenosti $log_2 p$, jer b^{p-2} možemo brzo potencirati.
Sveukupna složenost ovog rješenja je $O(2^K * lg_2 p)$, ili $O(2^K * K * lg_2 p)$ ako koristimo sporiju implementaciju.

Još nekoliko zadataka

1.) Modifikacija prošlog zadatka. Umjesto poteza jedan skok desno i jedan skok dolje, skokovi su u obliku slova L, kao što skače konj u šahu, s tim da se svakim skokom mora doći na polje koje je bliže cilju. Opet se traži broj različitih putova kojima konj može doći od gornjeg lijevog kuta do donjeg desnog kuta. (Google Code Jam 2008, zadatak Endless Knight)
2.) Koliko ima brojeva u intervalu [A, B], A <= B <= 10^{10} koji su djeljivi sa brojevima koji se sastoje samo od znamenaka 7 i 4.
Zadatak se zove TheAlmostLuckyNumbers, preuzet s Topcoder-a.
http://www.topcoder.com/ (c)2006, TopCoder, Inc.
3.) Zadatak žabe sa studentskog natjecanja 2009. godine.
http://www.hsin.hr/studenti2009/zadaci/zadaci.pdf
4.) Zadatak sweets sa srednjoeuropskog natjecanja 2004. godine.
http://www.oi.edu.pl/ceoi2004/problems/swe.pdf