Opisi algoritama

Primjeri implementiranih rješenja dani su u priloženim izvornim kodovima koji nužno ne odgovaraju u svim detaljima ovdje opisanim algoritmima.

Zadatak: Zbroj

autor: Adrian Satja Kurdija

Najmanji traženi broj dobit ćemo krećući od broja 100...00 i povećavajući znamenke s desne strane (budući da su to najmanje promjene) dok njihov zbroj ne postane jednak K.

Preciznije, točan je sljedeći algoritam: kreni od broja 10^{N-1} i sve dok je zbroj znamenaka manji od K, pronađi prvu znamenku s desne strane koja je manja od 9 i povećaj je za jedan. Ovaj algoritam može se ubrzati tako da znamenke zdesna odmah povećavamo na 9 dok nam do zbroja K ne ostane razlika manja od 9.

Najveći traženi broj dobit ćemo krećući od broja 999...99 i smanjujući znamenke s desne strane (budući da su to najmanje promjene) dok njihov zbroj ne postane jednak K.

Preciznije, točan je sljedeći algoritam: kreni od broja $10^N - 1$ i sve dok je zbroj znamenaka veći od K, pronađi prvu znamenku s desne strane koja je veća od 0 i smanji je za jedan. Ovaj algoritam može se ubrzati tako da znamenke zdesna odmah smanjujemo na 0 dok nam do zbroja K ne ostane razlika manja od 9.

Da bi algoritam bio učinkovit za $N \leq 10^6$ znamenaka, potrebno je broj održavati kao niz (listu) znamenaka, a ne kao cjelobrojni tip podataka (int).

Zadatak: Wordle

autor: Vedran Kurdija

Za svaku moguću riječ od pet slova provjeravamo zadovoljava li sve pokušaje te na kraju ispisujemo koliko riječi ih sve zadovoljava. Za generiranje svih mogućih riječi od pet slova koristimo, primjerice, pet ugniježđenih for petlji ili rekurzivnu funkciju.

Za provjeru zadovoljava li riječ neki pokušaj, pretpostavljamo da je ta riječ tražena te obojimo slova pokušaja shodno tome. Ako su boje jednake bojama koje je igra dala Mislavu, ta riječ zadovoljava taj pokušaj.

Riječ obojimo korištenjem pomoćnog niza xmz u koji za svako slovo engleske abecede $xmz(slovo)$ na početku zapisujemo odgovarajuću vrijednost $X - Z$, gdje X i Z odgovaraju oznakama iz teksta zadatka. Potom obilazimo slova u pokušaju te za svako koje nije zeleno, tj. koje se ne podudara sa slovom u traženoj riječi, provjeravamo je li $xmz(slovo) > 0$. Ako jest, boja tog slova je narančasta i smanjujemo vrijednost $xmz(slovo)$ za 1, a u suprotnom je boja tog slova siva. Za implementacijske detalje pogledajte priloženi kod.

Zadatak: Ispit

autor: Adrian Satja Kurdija

Da bismo smislili rješenje za bilo koji K, promotrimo najprije slučaj $K = 2$ koji je nosio ukupno 15 bodova. Broj traženih parova "potpuno različitih" natjecatelja možemo dobiti kao: broj svih parova, minus broj parova koji imaju jednak prvi zadatak, minus broj parova koji imaju jednak drugi zadatak, plus broj parova koji imaju jednaka oba zadatka (jer smo takve parove dvaput oduzeli).

Za $K = 3$ formula je malo složenija: nakon oduzimanja parova koji imaju isti prvi, pa drugi, pa treći zadatak, pribrajamo parove koji imaju jednaka dva zadatka (prvi i drugi, pa prvi i treći, pa drugi i treći) jer smo ih dvaput oduzeli, te na koncu ponovno oduzimamo parove koji imaju jednaka sva tri zadatka jer smo ih prethodno triput oduzeli i triput pribrojili.

Općenito, za proizvoljan K, ovo je zapravo formula uključivanja i isključivanja: oduzimamo parove koji se

podudaraju u podskupu s neparnim brojem zadataka, a pribrajamo one koji se podudaraju u podskupu s parnim brojem zadataka.

Ovo možemo implementirati tako da za svakog natjecatelja konstruiramo 2^K "signatura", od kojih se svaka sastoji od njegovih bodova za određeni podskup zadataka (npr. prvi, treći i četvrti zadatak). Za određeni podskup, odgovarajuće signature svih natjecatelja možemo sortirati i tako pronaći broj parova natjecatelja koji se u njima podudaraju, jer će jednake signature biti uzastopne u sortiranom nizu. Taj broj parova oduzimamo ili pribrajamo konačnom rezultatu ovisno o parnosti podskupa, tj. parnosti broja zadataka u signaturi.

Zadatak: Boje

autor: Vedran Kurdija

Promotrimo slučaj u kojem pokušaj ne sadrži dva ista slova. Rješenje za ovaj slučaj nosilo je 15 bodova. Jednom for petljom prolazimo po slovima pokušaja te ispisujemo 'Z' ako se trenutno slovo podudara s odgovarajućim slovom tražene riječi, a u suprotnom drugom for petljom prolazimo po svim slovima tražene riječi te ispisujemo 'N' ako među njima pronađemo trenutno slovo pokušaja, a u suprotnom ispisujemo 'S'.

Za sve bodove ovo rješenje nadograđujemo pomoćnim nizom xmz u koji za svako slovo engleske abecede $xmz(slovo)$ na početku zapisujemo odgovarajuću vrijednost $X - Z$, gdje X i Z odgovaraju oznakama iz teksta zadatka. Pri ispisu rješenja, za svako slovo u pokušaju koje nije zeleno, provjeravamo je li $xmz(slovo) > 0$. Ako jest, ispisujemo 'N' i smanjujemo vrijednost $xmz(slovo)$ za 1, a u suprotnom ispisujemo 'S'.

Zadatak: Znamenke

autor: Adrian Satja Kurdija

Broj N najprije nadopunimo vodećim nulama tako da ima 19 znamenaka, što je maksimalan broj znamenaka s obzirom na ograničenje $K \leq 170$. Zadatak rješavamo generirajući sve moguće kandidate za traženi broj X. Najprije, na sve moguće načine biramo prefiks zajedničkih znamenaka brojeva N i X, tj. koliko će se vodećih znamenaka (između 0 i 18) broja X podudarati s brojem N. Za svaki taj odabir, na sve moguće načine biramo iduću znamenku broja X (između 0 i 9) koja se razlikuje od odgovarajuće znamenke broja N. Ovisno o odabranoj znamenki, tj. o tome je li ona veća ili manja od odgovarajuće znamenke broja N, znamo hoće li promatrani kandidat X biti manji ili veći od N.

Kako odrediti ostale znamenke broja X? Ako je $X < N$, preostali dio broja X treba biti što veći da bismo se maksimalno približili broju N, pa stavljamo što veće znamenke (99...) dok ne dosegnemo zbroj znamenaka K, nakon čega broj X nadopunimo nulama. S druge strane, ako je $X > N$, preostali dio broja X treba biti što manji da bismo se približili broju N, pa X do kraja dopunjujemo nulama i onda povećavamo najmanje značajne znamenke (zdesna nalijevo) dok je zbroj znamenaka manji od K. Preciznije, nakon nadopune nulama, pronalazimo prvu znamenku s desne strane koja je manja od 9 i povećavamo je za jedan, dok zbroj znamenaka ne postane jednak K.

Na kraju od svih uspješno generiranih kandidata za X biramo onaj koji je najbliži broju N.

Zadatak: IMO

autor: Adrian Beker

Neka je K broj zadataka, a M maksimalan broj bodova po zadatku. U situaciji opisanoj u zadatku imamo $K = 6$ i $M = 7$. Rezultatom ćemo zvati proizvoljan niz koji se sastoji od K cijelih brojeva iz intervala $[0, M]$. Primijetimo da postoji $(M + 1)^K = 8^6 = 262144$ mogućih rezultata. *Minimum* dvaju rezultata $(a_i)_{i=1}^K$ i $(b_i)_{i=1}^K$ definiramo kao rezultat $(\min(a_i, b_i))_{i=1}^K$. Za niz $(a_i)_{i=1}^K$ reći ćemo da *dominira* niz $(b_i)_{i=1}^K$ ako je $a_i \geq b_i$ za svaki $1 \leq i \leq K$.

Trojke sa željenim svojstvom možemo prebrojiti tako da na sve moguće načine odaberemo tri različita natjecatelja te ispitamo je li minimum rezultata prve dvojice jednak rezultatu trećega. Taj pristup ima složenost $O(N^3 \cdot K)$ te je dovoljan za testne primjere u kojima vrijedi $N \leq 300$.

U slučaju da se rezultat svakog natjecatelja sastoji isključivo od brojeva 0 i M, opisani pristup možemo

ubrzati na sljedeći način. Rezultate pamtimo kao cijele brojeve u binarnom zapisu, pri čemu pojavljivanja broja M zamjenjujemo brojem 1. Za svaki mogući rezultat najprije prebrojimo koliko se puta pojavljuje u ulazu. Tada na sve moguće načine odaberemo tri rezultata takva da je treći jednak minimumu prvih dvaju te množenjem odgovarajućih brojača izračunamo koliko ima trojki natjecatelja koji imaju upravo te rezultate. Pritom treba paziti da brojimo samo one trojke u kojima su svi natjecatelji različiti. Budući da imamo svega 2^K mogućih rezultata, ovaj pristup ima složenost $O(N + 2^{3K} \cdot K)$. On se lako može ubrzati do složenosti $O(N + 2^{2K} \cdot K)$ ukoliko iskoristimo zamjedbu da prva dva odabrana rezultata jedinstveno određuju treći, no to nije bilo nužno.

Početni pristup u općenitom slučaju možemo ubrzati tako da, kada fiksiramo prva dva natjecatelja iz trojke, rješenju pribrojimo broj natjecatelja čiji je rezultat jednak minimumu rezultatu odabrane dvojice. Takav pristup podrazumijeva da smo prethodno za svaki mogući rezultat prebrojili koliko se puta on pojavljuje u ulazu. To možemo efikasno učiniti tako da rezultate promatramo kao cijele brojeve zapisane u sustavu s bazom $M+1$ te za svaki broj od 0 do $(M+1)^K - 1$ držimo odgovarajući brojač. Na taj način dobivamo algoritam složenosti $O(N^2 \cdot K + (M+1)^K)$, što je dovoljno za testne primjere u kojima vrijedi $N \leq 3000$.

Za sve bodove postupamo na obrnut način: na sve moguće načine odaberemo trećeg natjecatelja te brojimo na koliko načina možemo odabrati prva dva. U tu svrhu koristimo formulu uključivanja-isključivanja. Preciznije, za fiksnog trećeg natjecatelja C, označimo sa S skup svih uređenih parova (A, B) ostalih natjecatelja koji su na svakom zadatku ostvarili više ili jednako bodova nego C, tj. čiji rezultati dominiraju rezultat od C. Nadalje, za $1 \leq i \leq K$ označimo sa S_i skup parova iz S u kojima oba natjecatelja na i-tom zadatku imaju strogo više bodova nego C. Tada je traženi broj parova jednak $\left|\bigcap_{i=1}^{K} S_i^c\right|$, gdje je $S_i^c = S \setminus S_i$ komplement skupa S_i u skupu S. Prema formuli uključivanja-isključivanja imamo

$$\left|\bigcap_{i=1}^{K} S_i^c\right| = \sum_{I \subseteq \{1,\ldots,K\}} (-1)^{|I|} \left|\bigcap_{i \in I} S_i\right|,$$

stoga se problem svodi na određivanje kardinalnosti skupa $\bigcap_{i \in I} S_i$, za proizvoljan $I \subseteq \{1, \ldots, K\}$. Označimo rezultat natjecatelja C s $(c_i)_{i=1}^K$ te indikatorsku funkciju skupa I s 1_I ($1_I(i) = 1$ ako je $i \in I$ te $1_I(i) = 0$ inače). Tada je lako vidjeti da vrijedi $|\bigcap_{i \in I} S_i| = v(v-1)$, gdje je v broj natjecatelja različitih od C koji su na i-tom zadatku ostvarili barem $c_i + 1_I(i)$ bodova, za svaki $1 \leq i \leq K$.

Dakle, problem smo sveli na određivanje, za svaki mogući rezultat, koliko natjecatelja ima rezultat koji ga dominira. To možemo učiniti koristeći dinamičko programiranje i formulu uključivanja-isključivanja, po analogiji s uobičajenim načinom računanja dvodimenzionalnih prefiks-suma. Preciznije, za fiksan rezultat $r = (r_i)_{i=1}^K$, označimo s T skup natjecatelja čiji rezultat dominira r. Također, označimo s T_i skup natjecatelja iz T koji na i-tom zadatku imaju strogo više od r_i bodova. Tada je T disjunktna unija skupa natjecatelja koji imaju rezultat r i skupa $\bigcup_{i=1}^K T_i$. Po formuli uključivanja-isključivanja znamo da vrijedi

$$\left|\bigcup_{i=1}^{K} T_i\right| = \sum_{I \subseteq \{1,\ldots,K\},\ I \neq \varnothing} (-1)^{|I|-1} \left|\bigcap_{i \in I} T_i\right|.$$

Primijetimo da je $\left|\bigcap_{i \in I} T_i\right|$ jednako broju natjecatelja koji su na i-tom zadatku ostvarili barem $r_i + 1_I(i)$ bodova, za svaki $1 \leq i \leq K$. Iz tog se razloga prirodno nameće sljedeći pristup. Neka je $x = (x_i)_{i=1}^K$ niz cijelih brojeva u intervalu $[0, M+1]$ te označimo s $f(x)$ broj natjecatelja čiji rezultat dominira x. Ako x nije rezultat, odnosno sadrži broj $M+1$, tada je $f(x) = 0$. U suprotnome, iz gornjih zamjedbi slijedi

$$f(x) = g(x) + \sum_{I \subseteq \{1,\ldots,K\},\ I \neq \varnothing} (-1)^{|I|-1} f(x + 1_I),$$

gdje smo s $g(x)$ označili broj natjecatelja s rezultatom jednakim x. Dakle, vrijednosti funkcije f možemo izračunati dinamičkim programiranjem. Uz pažljivu implementaciju, ukupna složenost rješenja postaje $O(N + (M+1)^K \cdot 2^K)$. Za implementacijske detalje pogledajte priložene izvorne kodove.

Za kraj, spomenimo još da je zadatak moguće riješiti i u boljoj složenosti $O(N + (M+1)^K \cdot K)$, koristeći pristup sličan tzv. SOS (eng. *sum over subsets*) dinamici. Razradu tog rješenja ostavljamo čitateljici za vježbu.