

Elektrijada 2019

Algoritmi i strukture podataka - Rešenja

Zadatak 1

Za ovaj zadatak je potrebno razumevanje brojevnih sistema i prebacivanje brojeva iz jednog u drugi. Ovde prikazujemo jedan način kojim se može izračunati ono što se traži u zadatku, a da je pogodan za računanje na papiru, i primeri za uštedu vremena tamo gde je moguće.

Pošto u primeru pod a) broj nije veliki, prebacićemo ga u bazu 10:

$$(31034)_5 = 3 * 5^4 + 1 * 5^3 + 0 * 5^2 + 3 * 5^1 + 4 * 5^0 = (2019)_{10}$$

Brzo možemo uočiti da je 2^{10} najveći stepen dvojke manji od 2019, jer je $2^{10} = 1024 < 2019 < 2^{11} = 2048$

Što dalje znači da će ovaj broj u binarnom sistemu imati tačno 11 cifara.

Za primer pod b) potrebno je uočiti da se cifre iz primera pod a) ponavljaju dva puta, pa možemo koristeći prethodni rezultat brže izračunati broj u bazi 10.

$$(3103431034)_5 = (31034)_5 * (5^5)_{10} + (31034)_5 = (2019 * 3125)_{10} + (2019)_{10} = 6311394$$

Kako smo već videli da je $2^{11} = 2048$, brzo možemo proceniti da je $2^{22} = (2^{11})^2 = 2048^2 = \dots$ oko 4 miliona (već vidimo da je nepotrebno izračunati tačan broj, jer samo treba da uporedimo sa 6311394), a da je 2^{23} oko 8 miliona. Pa odatle zaključujemo $2^{22} < 6311394 < 2^{23}$, što znači da će ovaj broj kada ga prebacimo u binarni sistem imati 23 cifre.

Zadatak 2

Ukoliko bismo postpuno množili svaka dva broja i sabirali rezultate, broj ovakvih množenja je veliki, i samim tim i mogućnost da se napravi greška. Korisno je osmisliti algoritam koji ima što manje množenja, i gde proizvod tih množenja možemo lako računati.

Jedan način je da primetimo da se svaki broj množi sa svim drugima osim sa samim sobom, pa ako obeležimo elemente niza sa A_1, A_2, \dots, A_n , i sumu niza sa $S = \sum_{i=1}^n A_i$, imamo da je rezultat

(R) jednak:

$$\begin{aligned}
 2 * R &= A_1 * (A_2 + A_3 + \dots + A_n) + A_2 * (A_1 + A_3 + A_4 + \dots + A_n) + A_3 * (A_1 + A_2 + A_4 + \dots + A_n) + \\
 &\quad + \dots + A_n * (A_1 + A_2 + A_3 + \dots + A_{n-1}) \\
 &= A_1 * (S - A_1) + A_2 * (S - A_2) + \dots + A_n * (S - A_n) \\
 &= \sum_{i=1}^n A_i * (S - A_i) \\
 &= \sum_{i=1}^n A_i * S - \sum_{i=1}^n A_i^2 \\
 &= S * \sum_{i=1}^n A_i - \sum_{i=1}^n A_i^2 \\
 &= S^2 - \sum_{i=1}^n A_i^2
 \end{aligned}$$

(ovako dobijeni rezultat je potrebno podeliti sa 2 zbog toga što $A_1 * A_2$ i $A_2 * A_1$ treba uzeti samo jednom u razmatranje).

Sumu niza nije teško izračunati, a ukoliko znamo neki od “trikova” kako brzo računati kvadrate dvocifrenih brojeva, primer pod a) možemo brzo izračunati ovom metodom.

Za primer pod b) je potrebno primetiti da su elementi niza oblika $A_i = 1 + 7 * i$

Prikažaćemo kako da izračunamo sumu ovakvog niza, kao i sumu kvadrata njegovih elemenata. Koristićemo da je $\sum_{i=1}^n i = \frac{n*(n+1)}{2}$, i $\sum_{i=1}^n i^2 = \frac{n*(n+1)*(2n+1)}{6}$

$$S = \sum_{i=1}^n A_i = \sum_{i=1}^n (1 + 7i) = n + 7 * \sum_{i=1}^n i = n + 7 * \frac{n*(n+1)}{2}$$

Kako je u primeru $n = 20$, imamo da je $S = 20 + 7 * \frac{20*(20+1)}{2} = 1490$

Suma kvadrata elemenata (KS) je jednaka:

$$KS = \sum_{i=1}^n A_i^2 = \sum_{i=1}^n (1 + 7i)^2 = \sum_{i=1}^n (1^2 + 2 * 1 * 7i + (7i)^2) = \sum_{i=1}^n 1^2 + \sum_{i=1}^n 14i + \sum_{i=1}^n 7^2 i^2$$

$$= n + 14 * \frac{n*(n+1)}{2} + 49 * \frac{n*(n+1)*(2n+1)}{6}$$

$$\text{Za } n = 20, \text{ imamo da je } KS = 20 + 14 * \frac{20*(20+1)}{2} + 49 * \frac{20*(20+1)*(2*20+1)}{6} = 143590$$

$$\text{Konačno, imamo da je } R = (S^2 - KS) / 2 = (1490^2 - 143590) / 2 = 1038255$$

Zadatak 3

Ovo je jedan od osnovnih zadataka iz oblasti dinamičkog programiranja. Ovde ćemo pokazati i kako možemo ubrzati računanje za primere date u zadatku.

Ukoliko obeležimo sa (0,0) gornje levo polje, a (n,m) donje desno polje gde je n broj redova, a m broj kolona date matrice, i sa $D[i][j]$ označimo broj različitih puteva da stignemo do polja (i,j) onda možemo uvideti da je $D[i][j] = D[i-1][j] + D[i][j-1]$, a bazni slučaj je $D[0][0] = 1$.

Možemo redom popunjavati matricu, olakšavajuća okolnost je što treba ispisati samo poslednje 3 cifre rezultata, pa onda i za svaki međurezultat je dovoljno uvek pamtititi samo poslednje 3 cifre.

Za neke od datih primera možemo uočiti specifičnosti i olakšati računanje.

Ako imamo praznu matricu (bez blokirajućih polja) dimenzija (n,m) možemo videti da je broj puteva od gornjeg levog do donjeg desnog polja jednak $P[n][m] = \binom{n+m-2}{n-1}$

U prvom primeru je dovoljno od $P[n][m]$ oduzeti broj puteva koji prolaze kroz neki od data dva crna polja, a P možemo iskoristiti i za izračunavanje puteva koji prolaze kroz crna polja, recimo broj puteva koji prolaze kroz donje levo crno polje je $P[6][2] * P[2][4] = \binom{6}{1} * \binom{4}{1} = 6 * 4 = 24$

U drugom primeru možemo videti da moramo proći kroz centralno polje na mapi, a od početka do njega, kao i od njega do cilja su praktično prazne matrice, pa je rešenje u ovom primeru

$$P[5][6] * P[5][6] = \binom{9}{4}^2 = \left(\frac{9*8*7*6}{4*3*2}\right)^2 = 15876$$

Treći i četvrti primer se ne mogu posebno ubrzati osim da redom računamo vrednosti matrice D.

Zadatak 4

Potrebno je primetiti da ukoliko ispišemo sve anagrame neke reči, isto slovo će se pojaviti isti broj puta na svakoj poziciji u reči. Npr. za anagrame reči "BREG" slovo "B" bi se pojavilo 6 puta na prvom mestu, 6 puta na drugom mestu, 6 puta na trećem i 6 puta na četvrtom. Ovo znači da ukoliko se u zadatom primeru na nekoj poziciji pojavi 5 puta, to znači da se i u reči koja nedostaje nalazi baš na toj poziciji.

U primerima a) i b) sva slova u reči su različita, pa će i broj pojavljivanja za sva slova biti isti. Međutim u trećem se neka slova u reči ponavljaju, pa moramo pažljivo izračunati koliko će se puta koje slovo pojaviti na svakoj poziciji (npr. E će se više puta pojaviti na prvoj poziciji nego K). Takođe, da bismo iskoristili sortiranost niza za lakše brojeve, slova ćemo pronalaziti redom od prve pozicije.

Ako želimo da izračunamo koliko puta će se npr. slovo "K" pojaviti na prvoj poziciji u anagramima reči REKETER, to znači da prebrojimo koliko ima reči oblika Kxxxxxx gde na mesta "x" treba doći preostala tri slova "E", dva slova "R", i jedno slovo "T". To bi značilo da od 6 pozicija biramo 3 (za slovo E), pa zatim od preostale 3 pozicije biramo 2 (za slovo R) i na kraju će T biti na preostaloj poziciji. To je $\binom{6}{3} * \binom{3}{2} = 60$ kombinacija. Pažljivim brojanjem utvrdimo da

u datoj tabeli ima 53 reči u svakoj koloni (osim poslednje), i to će nam pomoći da lakše prebrojimo da ima ukupno 59 anagrama koji počinju slovom "K". Pošto smo već utvrdili da treba da ih bude 60, upravo onaj koji nedostaje mora počinjati slovom "K". Dalja slova pronalazimo na isti način, osim što posmatramo samo one reči koji počinju već sa "K", i onda anagrame preostalih slova "REETER".

Zadatak 5

U primeru pod a) možemo izračunati za svaki podniz redom koji bi bio XOR njegovih elemenata, pa zatim uzeti najveći od svih kao što se i traži, ali takav metod već u primeru pod b) postaje spor, a u primeru pod c) gotovo nemoguć da se izvede u računanju na papiru za vreme takmičenja. Zato ćemo pokazati algoritam kojim možemo brže doći do traženog maksimuma.

Da bismo lakše izračunali XOR podniza, možemo prvo izračunati XOR svih prefiksnih nizova, tj. XOR prva dva elementa niza, prva 3 elementa, itd. Ako sa $XPref[K]$ označimo XOR prvih K elemenata $XPref[K] = A[1] \text{ xor } A[2] \text{ xor } \dots \text{ xor } A[k]$, a sa $XPod[L][R]$ XOR podniza od L-tog do R-tog elementa $XPod[L][R] = A[L] \text{ xor } A[L + 1] \text{ xor } \dots \text{ xor } A[R]$, možemo uočiti da je

$$XPod[L][R] = XPref[R] \text{ xor } XPref[L - 1].$$

Ovo znači da ukoliko izračunamo ceo $XPref$ niz, treba naći samo dva elementa tog niza čiji je XOR maksimalan. Da bi XOR bio što veći, treba naći dva broja koja se razlikuju u najvrednijem bitu (prvom čitano s leva). Ukoliko ima više takvih parova, onda ih posmatramo sve zajedno, pa tražimo parove koji se razlikuju u drugom najvrednijem bitu, itd. dok ne stignemo do poslednjeg bita.