Opisi algoritama

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

Eventualna pitanja, mišljenja ili prijedloge vezane uz zadatke uputite na marinkisic14@gmail.com.

Zadatak: Tvrtka

Jedan od mnogo načina za rješiti ovaj zadatak jest u neko pomoćno 2D polje (*npr.* `zeli[sati_od][sati_do]`) pamtiti koliko radnika želi to radno vrijeme pa onda prema kriterijima iz zadatka odrediti rješenje.

Zadatak: Rim

Ideja je rekurzivno postaviti dvije domine. U jednom pozivu rekurzije probamo dominu postaviti gdje god možemo bilo vodoravno, bilo okomito. Kada rekurzija dođe na dubinu dva, povećamo rješenje za jedan. Zadatak se može rješiti i bez rekurzije, za detalje možete pogledati priložene kodove.

Zadatak: Proizvodi

Na početku treba parsirati input. Dakle proizvode označimo brojevima od 1 do N, a države brojevima od 1 do M, gdje su N i M broj različitih proizvoda, tj. država. Pretpostavimo da je $N \geq M$. Sada možemo napraviti matricu veličine NxN gdje na mjestu (i, j) piše cijena i-tog proizvoda u j-toj državi. Ako taj proizvod ne postoji, tada u matrici piše -1. Primjetimo sada da zbog uvijeta iz zadatka, Sanja će iz svakog retka i svakog stupca odabrati točno jedan proizvod. S obzirom na mala ograničenja, to nas dovodi do sljedećeg rješenja. Uzmimo neku permutaciju veličine N. Neka onda predtsavlja koje će proizvode Sanja kupiti, tj. za svaki i, Sanja će kupiti i-ti proizvod u `perm[i]` državi ako on postoji. Za konačno rješenje možemo isprobati sve permutacije i uz pomoć one koja odgovara kriterijima zadatka ispisati rješenje.

Zadatak: Radnici

Ideja rješenja je ista kao u zadataku Tvrtka, samo se traži malo drugačiji output te dodatan uvijet.

Zadatak: Euro

Za svaki proizvod zapamtimo najnižu cijenu i u kojoj državi ima tu cijenu. Spremimo trojke $(drzava, cijena, proizvod)$ u vektor, sortiramo ih te ih ispišemo u traženom formatu.

Zadatak: Putovanje

Ovaj zadatak rješit ćemo dinamičkim programiranjem. Stanje će nam se sastojati od dva broja. Prvi će označavati zadnji grad koji smo posjetili, dok će drugi predstavljati bitmasku koje smo sve gradove posjetili. Jedinica na i-tom mjestu te bitmaske će značiti da smo posjetili i-ti grad, dok će 0 značiti da nismo posjetili i-ti grad. U prijelazu, probamo za sljedeći grad koji ćemo posjetiti odabrati neki od gradova koje nismo posjetili. Složenost ovog rješenja jest $O(2^N * N^2)$.

Za one koji žele znati više, ovaj problem poznat je pod imenom *Traveling salesman problem* i više o njemu možete pročitati na *linku*.