

## 1. Programmēšanas darbs

### Uzdevums

Vinnija Pūka mežā ir celiņu tīkls; šo tīklu apraksta neorientēts grafs  $G = (V, E)$ , kur grafa šķautnes atbilst celiņiem un virsotnes celiņu krustojumiem. Grafam  $G$  ir  $n$  virsotnes (virsotņu kopa  $V = \{1, \dots, n\}$ ) un  $m$  šķautnes, grafs ir sakarīgs – no jebkura celiņu krustojuma ir iespējams nokļūt jebkurā citā, kā arī ir izbūvēti tiltiņi, kas ļauj celiņiem šķērsot vienu otru arī ārpus krustojumiem (tātad, grafs var arī nebūt planārs).

Pūku uztrauc tas, ka reizēm dodoties pastaigāties viņš aizmirst paņemt līdz medus podiņu, ar ko ceļā iestiprināties, tādēļ viņš ir nolēmis dažu celiņu malās ierīkot medus krājumus. Katra celiņa malā (apmēram tā vidusposmā) aug koks ar palielu dobumu, un Pūkam šie koki liekas ļoti piemēroti krājumu ierīkošanai. Taču piekļūšana visiem dobumiem nav vienlīdz viegla, līdz ar ko katra dobuma “grūtības pakāpi” Pūks ir novērtējis ar veselu skaitli intervālā no  $-100$  līdz  $100$ .

Pūka nolūks ir izvietot krājumus dažos no šiem kokiem tā, lai izpildītos sekojoši nosacījumi:

1. Katrā cikliskā maršrutā, kas iziet no kāda krustojuma (virsotnes)  $v$  un atgriežas tajā pat krustojumā  $v$ , un kurā visi ietilpstošie celiņi (šķautnes) ir dažādi, atrodas vismaz viens celiņš ar koka dobumu, kurā izvietoti medus krājumi.
2. No visiem krājumu izvietojumiem, kas atbilst pirmajam nosacījumam, visu to dobumu, kuros izvietots medus, grūtības pakāpju summa ir vismazākā iespējamā.

Izdomāt un implementēt pēc iespējas efektīvāku algoritmu, kas dotam grafam  $G = (V, E)$  (celiņu tīklam) atrod šķautņu (celiņu) kopu, pie kurām izvietojot medus krājumus tiek garantēts, ka abi Pūka izvirzītie nosacījumi izpildās.

### Tehniskie pieņēmumi un prasības

Programmu var rakstīt jebkurā tradicionālā programmēšanas valodā, kas daudzmaz nodrošina algoritma lasāmību un iespēju novērtēt programmas ātrdarbību (līdz ar ko nevar lietot, piemēram, ārējās bibliotēkas tādu aprēķinu veikšanai, kas prasa vairāk nekā konstantu izpildes laiku, nederēs arī funkcionālās programmēšanas valodas). Jāiesniedz:

- programmas teksts (pietiek ar failu, izdruka nav nepieciešama);
- izpildāms programmas fails;
- saturīgs algoritma darbības apraksts, pamatojums tam, ka algoritms izdos pareizu rezultātu, un algoritma sarežģītības novērtējums.

Ieteicamais darba iesniegšanas veids – iesūtīšana pa e-pastu (programmas kods, izpildāmais fails un darba apraksts). Iesniegtais darbu var būt nepieciešams demonstrēt klātienē un sniegt skaidrojumus par programmas tekstu un/vai aprakstu (tas atkarīgs no programmas izstrādes un programmēšanas valodas, algoritma apraksta precizitātes un saprotamības, un/vai citiem apstākļiem – ja demonstrācija būs nepieciešama, par to vienosimies individuāli).

# ĀTRU ALGORITMU KONSTRUĒŠANA UN ANALĪZE

## 2023. GADA RUDENS

**Ieejas dati.** Programmai jāspēj apstrādāt jebkuru ieejas failu, kas satur veselu skaitļu virkni:

$n \quad a_1 \quad b_1 \quad w_1 \quad a_2 \quad b_2 \quad w_2 \quad \dots \quad a_m \quad b_m \quad w_m,$

kur  $n$  – virsotņu skaits grafā (var pieņemt, ka  $n < 500$ ),  $m$  – šķautņu skaits grafā (ieejas datos tiešā veidā netiek norādīts),  $a_i, b_i \in \{1, \dots, n\}$ ,  $w_i \in \{-100, \dots, 100\}$ , un katrs trijnieks  $a_i, b_i, w_i$  apraksta šķautni starp virsotnēm  $a_i$  un  $b_i$ , kā arī šķautnei atbilstošā dobuma grūtības pakāpi  $w_i$ . Šķautnes un to galapunkti šajā sarakstā var parādīties brīvi izvēlētā secībā (tajā skaitā, viena un tā pati šķautne var tikt aprakstīta vai nu ar trijnieku  $a_i, b_i, w_i$ , vai arī ar trijnieku  $b_i, a_i, w_i$ ).

Kā atdalītāji starp skaitļiem var tikt lietotas "space" (0x20), "tab" (0x09) un "newline" (0x0D|0x0A vai 0x0A) simbolu virknes jebkurā garumā, kas lielāks vai vienāds ar 1, nenoteikta garuma atdalītājsimbolu virknes atļautas arī faila sākumā un beigās (vērtējums tiks samazināts, ja programmai būs papildus ierobežojumi uz atdalītājsimbolu lietojumu un tā nespēs apstrādāt kādu no testa ieejas failiem). Var toties pieņemt, ka ieejas datos nav kļūdu, un, ka dotais grafs ir sakarīgs.

**Rezultāts.** Programmai ir jāizdod izejas fails, kas satur skaitļu virkni:

$k \quad W \quad a_1 \quad b_1 \quad a_2 \quad b_2 \quad \dots \quad a_k \quad b_k,$

kur  $k$  ir dobumu skaits, kuros jāizvieto medus podiņi,  $W$  – šo dobumu grūtības pakāpju summa, kam seko pāru  $a_i, b_i$  saraksts (kopskaitā  $k$ ), kuri apraksta šķautnes, pie kurām jāizvieto krājumi (šķautņu un to galapunktu norādes secība nav svarīga; par atdalītājiem drīkst izmantot jebkuras atdalītājsimbolu virknes, kas atļautas ieejas failā).

**Darba vērtējums.** Lai darbs tiktu ieskaitīts, algoritmam ir jādarbojas polinomiālā laikā (atkarībā no  $n$  un  $m$ ) un darba aizstāvēšanā ir jādemonstrē izpratne par iesniegto programmu un aprakstu. Algoritma sarežģītībai jābūt polinomiālai, konkrētā risinājuma sarežģītība zināmā mērā var ietekmēt darba vērtējumu, taču tai nav jābūt labākajai iespējamajai, lai saņemtu maksimālo novērtējumu. 80% no atzīmes dos programmas, atlikušos 20% tās apraksta vērtējums.

