# Vinnijs Pūks

## Problēmas apraksts

Lācītim Pūkam ciemos pie sava drauga Sivēna ir iespējams nokļūt pa vairāku celiņu tīklu; šo tīklu apraksta neorientēts grafs G = (V,E), kur grafa šķautnes atbilst celiņiem un virsotnes celiņu krustojumiem. Grafam G ir pavisam n virsotnes v0, v1, …, vn−1, tiek pieņemts, ka lācīša Pūka mājiņa atrodas virsotnē v0 un Sivēna mājiņa atrodas virsotnē vn−1. Pūku uztrauc tas, ka reizēm dodoties ciemos 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 vidus posmā) aug koks ar palielu dobumu, un Pūkam šie koki liekas ļoti piemēroti krājumu ierīkošanai. Pūka nolūks ir izvietot krājumus dažos no šiem kokiem tā, lai neatkarīgi no tā, pa kādu ceļu viņš dotos pie Sivēna, viņam iznāktu iet garām vismaz vienam no kokiem, kurā atrodas medus krājumi. Tajā pašā laikā, Pūks ir pietiekami laisks, lai gribētu izveidot krājumus tik maz vietās, cik vien iespējams. Izdomāt un implementēt pēc iespējas efektīvāku algoritmu, kas dotam grafam G = (V,E) (celiņu tīklam) atrod mazāko iespējamo šķautņu (celiņu)

kopu, pie kurām izvietojot medus krājumus tiek garantēts, ka katrs ceļš no virsotnes v0 uz virsotni vn−1 saturēs vismaz vienu šķautni, pie kuras ir izvietoti

medus krājumi (ja ir vairākas šķautņu kopas ar prasīto īpašību, pietiek atrast jebkuru no tām).

**Ieejas dati**. Programmai jāspēj apstrādāt ieejas failu, kas satur naturālu skaitļu virkni n a0 b0 a1 b1 … am-1 bm-1, kura atbilst grafam ar n virsotnēm V0, V1, …, Vn−1 un m šķautnēm {Va0,Vb0}, {Va1,Vb1}, …, {Vam-1,Vbm-1}. Šķautnes un to galapunkti šajā sarakstā var parādīties brīvi izvēlētā kārtībā. Kā atdalītāji starp skaitļiem tiek lietotas "space" (0x20) un "newline" (0x0d + 0x0a vai 0x0a) simbolu virknes garumā, kas lielāks vai vienāds ar 1. Var pieņemt, ka dotais grafs ir sakarīgs (līdz ar ko vismaz viens ceļš no V0, uz Vn-1, vienmēr eksistē).

**Rezultāts**. Programmai ir jāizdod naturālu skaitļu virkne k a0 b0 a1 b1 … ak-1 bk-1, kur k ir mazākais iespējamais apjoms šķautņu kopai ar prasīto īpašību un {Va0,Vb0}, {Va1,Vb1}, …, {Vak-1,Vbk-1} ir viena no iespējamajām k šķautņu kopām ar prasīto īpašību. Precīzs izejas datu formāts nav svarīgs (tajā skaitā, var būt arī grafisks).

## Risinājuma apraksts

Lai atrastu iespējami mazāko celiņu skaitu un celiņus, kuru malā esošos kokos Pūkam ievietot medus krājumus, diezgan parocīgi ir izmantot *breadth-first search* algoritmu visu taciņu (celiņu kopu) no Pūka mājiņas līdz Sivēna mājiņai atrašanai, paralēli piefiksējot, cik reižu pa katru celiņu bijis jāiet, un beigās no katras taciņas celiņiem izvēlēties vienu celiņu, pa kuru kopumā ir bijis jāiet visvairāk.

Programmas kods rakstīts valodā JAVA. Lai palaistu Windows videi paredzēto programmu, kā parametrs jāpadod apstrādājamais teksta fails, piemēram,

* puks.exe src/ex1.txt
* puks.exe src/ex2.txt
* puks.exe src/ex3.txt

## Algoritma soļi:

1. Paņem Pūka mājiņai tuvākos krustojumus
   1. Katram krustojumam pārbauda, vai tas gadījumā nav Sivēna mājiņa
      1. Ja ir Sivēna mājiņa
         1. Pievieno to aktuālajai taciņai
         2. Saglabā aktuālo taciņu visu taciņu sarakstā
            1. Katru taciņas celiņu vai nu pievieno celiņu sarakstam ar vērtību 1, vai, ja celiņš jau ir sarakstā, palielina tā vērtību par 1
      2. Ja nav Sivēna mājiņa
         1. Pievieno to aktuālajai taciņai
         2. Izsauc 1. Ar aktuālo krustojumu Pūka mājiņas vietā
2. Apstaigā katru taciņu
   1. Apstaigā taciņas celiņus
      1. Katram celiņam pievieno vērtību
   2. Rezultātu sarakstam pievieno celiņu ar vislielāko vērtību (ja tā vēl nav sarakstā)
3. Izdrukā rezultātu saraksta izmēru un katru tajā esošo celiņu

## Sarežģītības novērtējums

Tā kā *breadth-first search* algoritms ir novērtējams ar O(E+V) un šis algoritms, papildus *breadth-first search*

algoritmam, vēl apstaigā visus celiņus jeb grafa šķautnes, tad šis algoritms novērtējams ar O(2E+V).