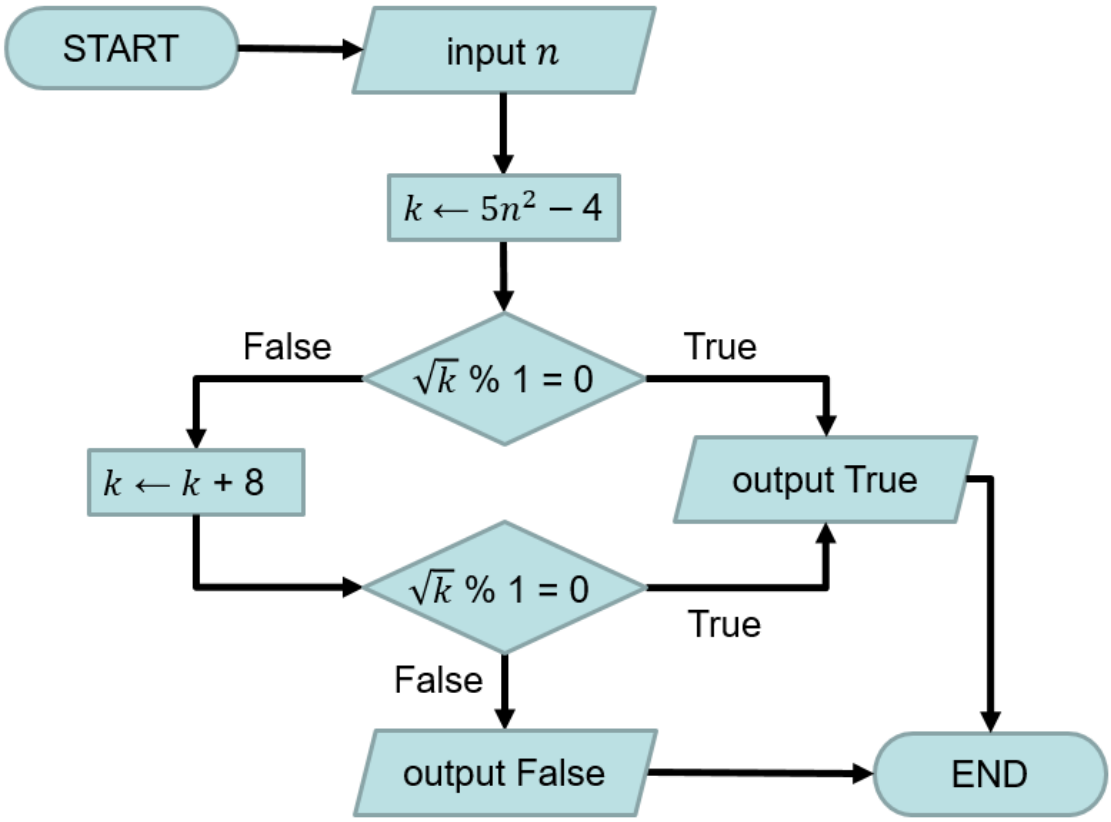
Dalyko „Algoritmai ir duomenų struktūros“ 2019 m. egzamino užduočių sprendimai

1. Vieno iš galimų algoritmų, patikrinančių, ar natūralusis skaičius n priklauso Fibonačio sekai, pseudokodas:

```
1. isFibonacciNumber(n):
2.     k ← 5n² − 4
3.     if √k % 1 = 0 then
4.         return(True)
5.     else
6.         k ← k + 8
7.         if √k % 1 = 0 then
8.             return(True)
9.         else
10.            return(False)
```

To paties algoritmo blokinė schema:

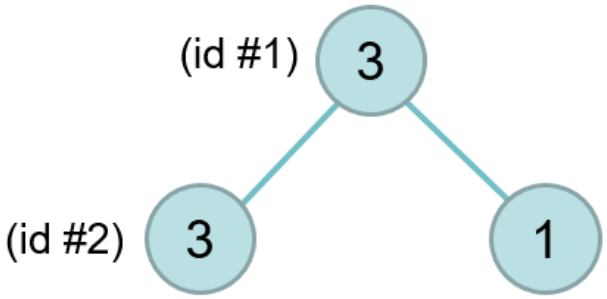


2. Taikant Euklido algoritmą, skaičių 408 ir 762 didžiausias bendras daliklis apskaičiuojamas šiuo principu:

$$762 = 408 \times 1 + 354$$
$$408 = 354 \times 1 + 54$$
$$354 = 54 \times 6 + 30$$
$$54 = 30 \times 1 + 24$$
$$30 = 24 \times 1 + \mathbf{\underline{6}}$$
$$24 = 6 \times 4 + 0$$

   Atsakymas: 6.

3. Išrikiuokime *HeapSort* algoritmu tokį sąrašą [3 (id #1), 3 (id #2), 1]. Šis sąrašas tenkina *MaxHeap* sąlygą:



   Esant teisingai *MaxHeap* sąlygai, medžio šaknies reikšmė įrašoma rikiuojamo sąrašo gale, todėl, realizavus *HeapSort* algoritmą, gaunamas toks išvesties sąrašas:

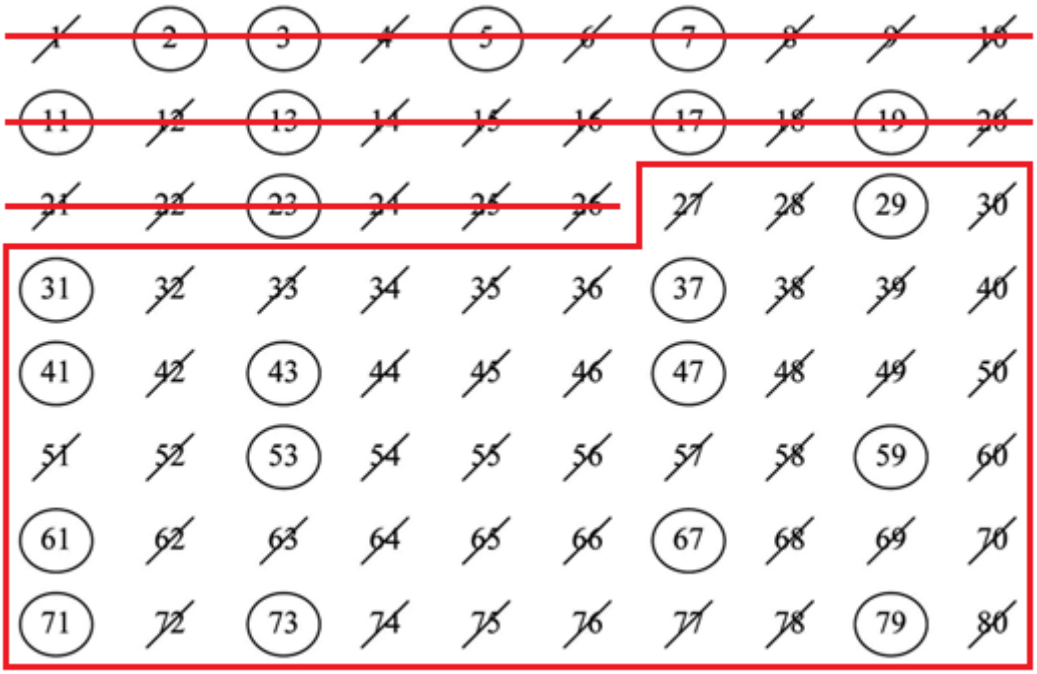
   [1, 3 (id #2), 3 (id #1)].

   Matome, kad po rikiavimo nebuvo išlaikytas vienodų elementų tarpusavio eiliškumas, todėl šis pavyzdys rodo, kad *HeapSort* algoritmas nėra stabilus.

4. Mažiausias keturženklis trejetainis skaičius lygus 1000, didžiausias – 2222. Paverskime šiuos skaičius į dešimtainius:

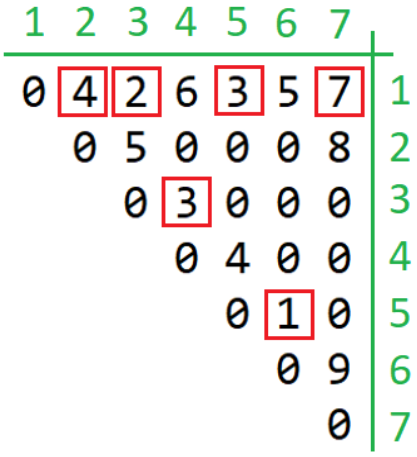
$$1000_3 = 1 \times 3^3 + 0 \times 3^2 + 0 \times 3^1 + 0 \times 3^0 = 27,$$
$$2222_3 = 2 \times 3^3 + 2 \times 3^2 + 2 \times 3^1 + 2 \times 3^0 = 80.$$

   Išvada. Taikant Eratosteno rėčio algoritmą reikia rasti visus dešimtainius pirminius skaičius, priklausančius intervalui [27; 80]:

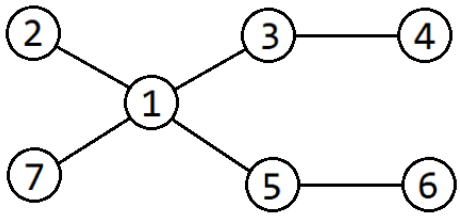


Atsakymas: 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79.

5. Pasinaudodami duota svorinio grafo gretimumo matrica bei taikydami Kruskalio algoritmą sudarykime minimalų jungiantįjį medį.



Į formuojamą medį, pagal mažiausius briaunų svorius 1, 2, 3, 3, 4, 7 (nesudarant ciklo), briaunos įterpiamos atitinkamai šia tvarka: {5, 6}, {1, 3}, {1, 5}, {3, 4}, {1, 2}, {1, 7}.



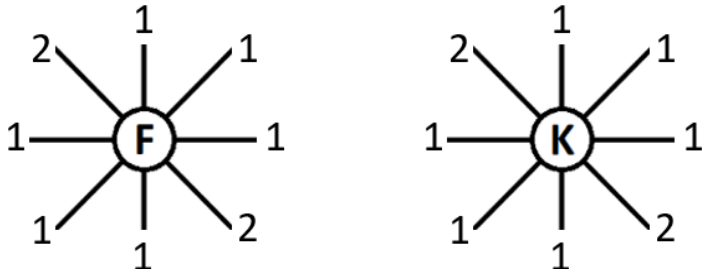
Gauto medžio Priuferio kodas α = [] suradomas iš medžio pašalinant briaunas **šia tvarka**:

{1, 2}: α = [1], **{3, 4}**: α = [1, 3], **{1, 3}**: α = [1, 3, 1],

{5, 6}: α = [1, 3, 1, 5], **{1, 5}**: α = [1, 3, 1, 5, 1].

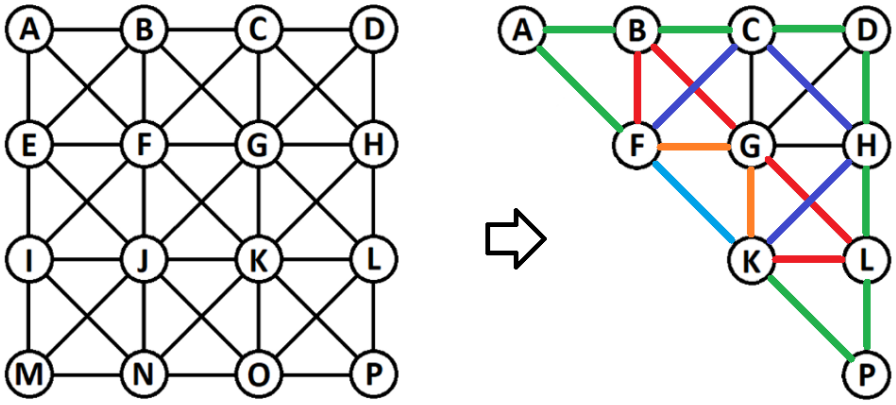
Atsakymas: α = [1, 3, 1, 5, 1].

6. Nesunku pastebėti, kad iš viršūnės F didžiausias galimas išeinantis srautas lygus 10 ir į viršūnę K didžiausias galimas patenkantis srautas – irgi 10:



Išvada: teoriškai maksimalus srautas negali viršyti 10, todėl, jei pavyktų tokį srautą apskaičiuoti, atsakymas būtų teisingas.

Pasinaudodami simetrija, padalinkime pradinį tinklą du potinklius, kuriuose visų briaunų talpa lygi 1. Vieno iš potinklių pavyzdys:



Šio potinklio maksimalus srautas lygus 5, nes nukeliauti iš **F** į **K** galima 5 skirtingais takais, kurių kiekvienas praleidžia srautą, lygų 1:

FABCDHLP**K**, **F**BGL**K**, **F**CH**K**, **F**G**K**, **FK**.

Kito simetrinio potinklio maksimalus srautas analogiškai lygus 5, todėl viso tinklo maksimalus srautas lygus 10.

7. Tegu dėlionės detalių skaičius lygus $k = m \times n$. Dėlionės dėjimo principas yra analogiškas išrinkimo (*SelectionSort*) algoritmui, nes, kaskart paėmus detalę, pačiu nepalankiausiu atveju reikia pereiti visas galimas neužimtas padėtis, t. y. $4k$, $4(k-1)$, $4(k-2)$, … iki kol detalė „atranda“ savo vietą. Viso daugiausiai galimų žingsnių yra

$$\sum_{i=0}^{k-1} 4(k-i) = 4\sum_{i=1}^{k} i = 4 \cdot \frac{k(k-1)}{2} = 2k^2 - 2k.$$

Todėl nagrinėjamo algoritmo sudėtingumas lygus $O(k^2) = O(m^2n^2)$.