DUOMENŲ STRUKTŪROS IR ALGORITMAI

MARIUS GŽEGOŽEVSKIS

KONTAKTINĖ INFORMACIJA

E-mail: m.gzegozevskis@eif.viko.lt

Moodle: Algoritmai ir duomenų struktūros, prisijungimo raktas = **labas**.

Auditorija: 103

KURSO SANDARA

ĮVADAS Į DUOMENŲ STRUKTŪROS IR ALGORITMAI KURSĄ.

DUOMENŲ STRUKTŪROS (Stekas, Eilė, Masyvas, Medis ir kitos).

RŪŠIAVIMO ALGORITMAI (QuickSort, BogoSort, BubbleSort ir kt.)**.**

PAIEŠKOS ALGORITMAI.

ALGORITMŲ ANALIZĖ.

LITERATŪRA

- A. Juozapavičius. **Duomenų struktūros ir algoritmai.** Vilnius, VU. 1997. (PDF)
- Daliaus Mažeikos **„Duomenų struktūros ir algoritmai“.** (LINK)
- Vladas TUMASONIS, **Informatika 2014 konspektas**. (LINK)
- **Problem Solving with Algorithms and Data Structures**, Brad Miller, David Ranum. (PDF)
- **Data Structures and Algorithms in Python** Michael T. Goodrich Department of Computer Science. (PDF)
- **Problem Solving with Algorithms and Data Structures** by Brad Miller and David Ranum, Luther College – Internete patalpintas knygos turinys + Python kodo interpretatorius online. (LINK)
- Algorithms + Data Structures = Programs Niklaus Wirth Prentice-Hall 1975 ISBN 0-13-022418-9 366 pages,

figures

VERTINIMO KRITERIJAI

Egzaminas (**E**).

Savarankiška užduotis (**SU**) .

Praktinių užsiėmimų užduotys (**PU**).

Galutinis įvertinimas = E * 0.3 + SU * 0.4 + PU * 0.3;

EGZAMINAS

Egzaminas testo forma su atviraisiais klausimais.

Egzaminas sudarytas iš paskaitų metu išdėstytos teorijos.

Pavyzdžiui pateiktas programos kodo fragmentas ir nustatyti algoritmo asimptotinę notaciją.

Koks yra algoritmo sudėtingumas O(n), O(log n) ir t.t.

Kas yra algoritmas, kas yra pseudokodas. Arba duota algoritmo realizacija nubraižyti blokinę schemą, užrašyti pseudokodą ir panašaus pobūdžio klausimai.

SAVARANKIŠKA UŽDUOTIS

1. Sugalvoti probleminę sritį. (Atsiųsti el. paštu: m.gzegozevskis@eif.viko.lt , **Subject:** PI14A, B ar C, užduoties pavadinimas). **Terminas:** pirmas semestro mėnuo.

2. Pasiūlyti naują algoritmą iš bet kurios srities (Rūšiavimas, trumpiausio kelio radimas, ekspertinė sistema ir kt.) arba esamųjų algoritmų modifikavimas, apjungimas, pritaikymas jūsų siūlomai problemai spręsti.

Minimalūs algoritmo aprašo reikalavimai:

- ESMINĖ IDĖJA.
- PSEUDOKODAS.
- ALGORITMO SCHEMOS.
- REALIZACIJA. **Python, C/C++, Pascal programavimo kalba.**

ATSISKAITYMAS. Praktinės paskaitos metu pademonstruoti, jūsų realizuotą algoritmą, rezultatai, tarpiniai skaičiavimai ir kita svarbi informacija. Taip pat sugebėti pakomentuoti jūsų atlikto algoritmo veikimą, galbūt jeigu įmanoma atlikti vieną ar kitą modifikaciją.

PLAČIAU APIE REIKALAVIMUS Nuotolinio mokymo aplinkoje MOODLE. Kas neturi prieigos kreiptis el. paštu: m.gzegozevskis@eif.viko.lt arba į kolegas dėl užduoties reikalavimų dokumento.

PRAKTINIŲ UŽSIĖMIMŲ UŽDUOTYS

Praktinių užsiėmimų metu, bus atliekamos nesudėtingos užduotys, kurios palaipsniui sunkės už kurias gausite balus. Užduočių realizacija atliekama programavimo kalba **Python.**

DUOMENŲ STRUKTŪROS IR ALGORITMAI

Duomenų struktūros ir algoritmai - tai sritis, formalizuojanti platų spektrą svarbių ir vis labiau plintančių procedūrų, kompiuteriu sprendžiant simbolinių skaičiavimų, inžinerijos, kompiuterinės grafikos, duomenų bazių, telekomunikacijų ir kitų sričių uždavinius.

Ji jungia įvairius matematinės logikos, kombinatorikos, diskrečiosios matematikos, programavimo, kompiuterinės technikos konstravimo, geometrijos ir algebros metodus.

DUOMENŲ STRUKTŪROS IR ALGORITMAI

Duomenų struktūra (**DS**) - tai duomenų aibė, kuriai apibrėžtos tam tikros operacijos ir loginiai ryšiai tarp jų. Literatūroje naudojamas ir kitas pavadinimas (dažnai suprantamas kaip sinonimas) - abstraktūs duomenų tipai (ADT).

Duomenų struktūros esmė informatikos požiūriu yra tokia:

- **apibrėžti vietos išskyrimo duomenims talpinti kompiuterio atmintyje metodą;**
- **apibrėžti duomenų reikšmių rašymo į šią vietą metodą.**

PAGRINDINIAI DUOMENŲ STRUKTŪRŲ OBJEKTAI

- Duomenų ir algoritmų aprašymo (ir užrašymo) kalba.
- Baziniai duomenų tipai ir elementarios duomenų struktūros.
- Duomenų struktūrų ir atitinkamų algoritmų analizės ir diegimo metodai.

BAZINIAI DUOMENŲ TIPAI

Duomenų struktūros, įdiegtos programavimo kalboje.

Paskalio kalboje tai yra: sveikas (**integer**), realus (**real**), loginis (**boolean**), simbolinis (**character**), masyvas (**array**), aibė (**set**), rodyklės tipas (**pointer**), įrašo tipas (**record**).

Užrašydami visas kitas duomenų struktūras ar algoritmus konkrečioje realizacijoje (**programoje**), turime jas išreikšti baziniais tipais ir tam tikromis operacijomis su jais.

ELEMENTARIOS DUOMENŲ STRUKTŪROS

Elementarios duomenų struktūros - struktūros, labai paplitę ir nesudėtingai apibrėžiamos.

Jos gali būti realizuotos įvairiais lygiais: techniniu lygiu (operacijų vykdymas įdiegtas elektroninėje schemoje ar kitoje techninėje kompiuterio dalyje), programavimo kalba (realizuotos kalboje baziniais tipais), algoritmu (duomenų struktūros operacijos išreiškiamos algoritmiškai baziniais tipais).

MASYVAS

Masyvas - tai vieno tipo tolydžiai išdėstytų kintamųjų aibė, kai kintamųjų skaičius iš anksto yra žinomas, o kiekvieną kintamąjį galima išrinkti tiesiogiai (naudojant indeksą ar indeksus).

Toks apibrėžimas pateikia masyvą kaip bazinį tipą (pvz., Paskalio kalboje).

Duomenų struktūros atveju masyvas apibrėžiamas kaip fiksuotas kiekis duomenų, kurie atmintyje laikomi nuosekliai, o išrenkami naudojant indeksą ar indeksus, be to, kiekvieno duomens reikšmės formuojamos unifikuotai.

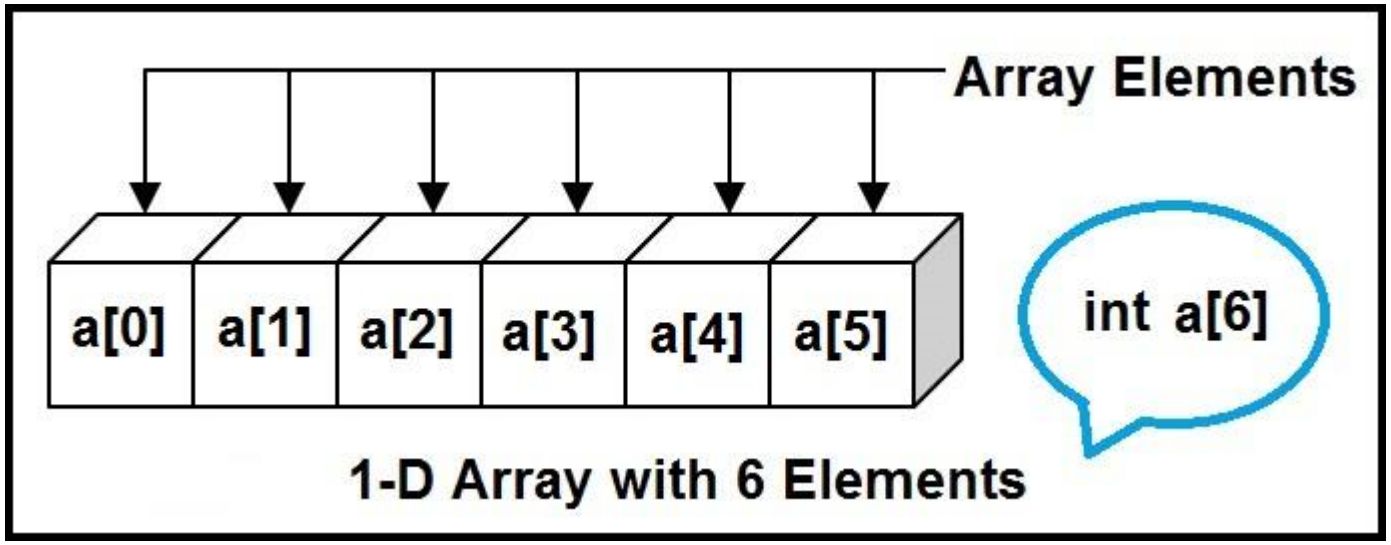
MASYVAS

Masyvas yra statinė duomenų struktūra - jo dydis turi būti žinomas iš anksto.

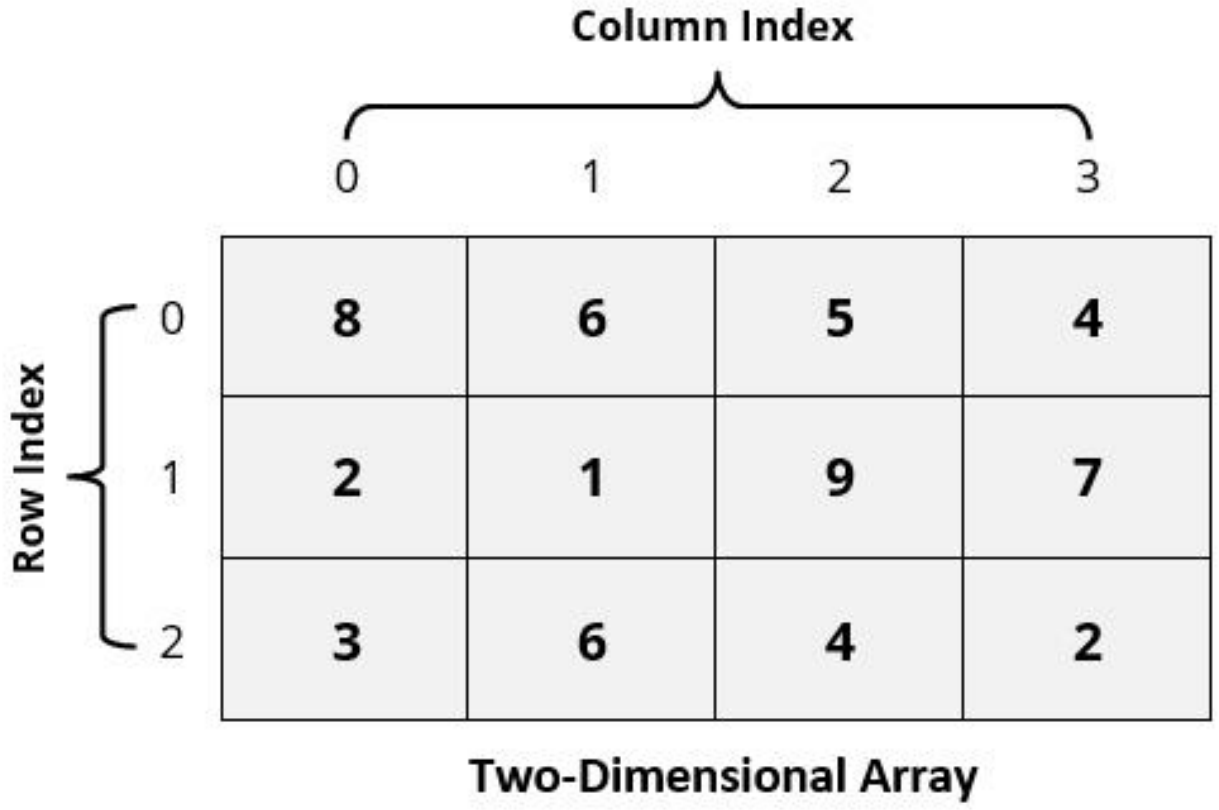
Masyvo paplitimą sąlygoja ir tai, kad faktiškai bet kurio kompiuterio atmintį galima interpretuoti kaip masyvą - indeksų vaidmenį vaidina adresai.

Duomenims išrinkti masyve gali būti naudojama ir daugiau kaip vienas indeksas - tada kalbama apie dvimačius, trimačius ir t. t. masyvus.

VIENMATIS MASYVAS



DVIMATIS MASYVAS



MASYVAS

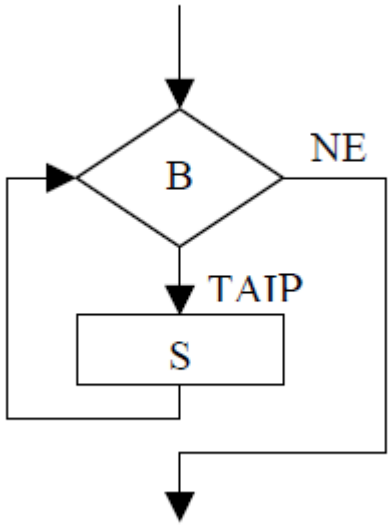
Išrinkimo laikas masyve bet kuriam duomeniui yra fiksuotas. Jis priklauso tik nuo masyvo matavimų skaičiaus, t.y. nuo formulės, kuria naudojantis pagal kelių indeksų reikšmes skaičiuojamas duomens saugojimo atmintyje adresas.

ALGORITMO SĄVOKA

Algoritmas – nurodymų seka, kurią atlikus su kažkokiais objektais, gaunamas uždavinio sprendinys.

$$S = \sum_{i=1}^{n} a_i ;$$

While (!sorted(Array))
shuffle (Array)



ALGORITMO SAVYBĖS

Masiškumas. Vienu algoritmu išsprendžiama daug uždavinių. **Rezultatyvumas.** Algoritmas turi duoti kokį nors rezultatą – tai gali būti ne tik lygties sprendiniai, bet ir, pvz, atsakymas, kad sprendinių nėra arba pradiniai duomenys klaidingi.

Efektyvumas. Nustatomas, taikant algoritmų analizės metodus.

Aiškumas. Algoritmas pateikiamas vykdytojui suprantama ir aiškia kalba.

Diskretumas. Algoritmą sudaro diskreti veiksmų seka. **Baigtinumas.**

ALGORITMO PASKIRTIS

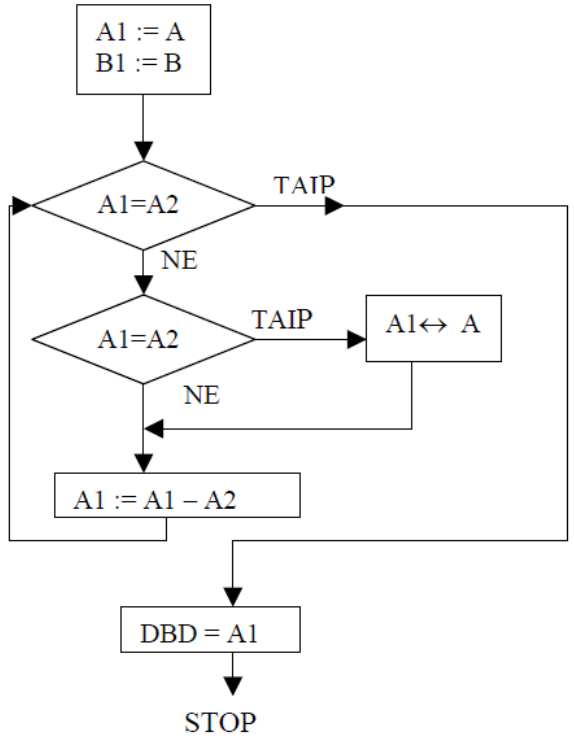
Su algoritmo sąvoka betarpiškai yra susijusi algoritmo **vykdytojo sąvoka**: algoritmas kuriamas tam, kad kažkas atliktų algoritmo nurodomus veiksmus.

Tas kažkas ir yra algoritmo vykdytojas. Sudarius ir užrašius algoritmą, jis gali būti atliekamas formaliai.

Taigi, jo atlikimą galima pavesti automatui-kompiuteriui. Šitaip kompiuteris tampa algoritmo vykdytoju.

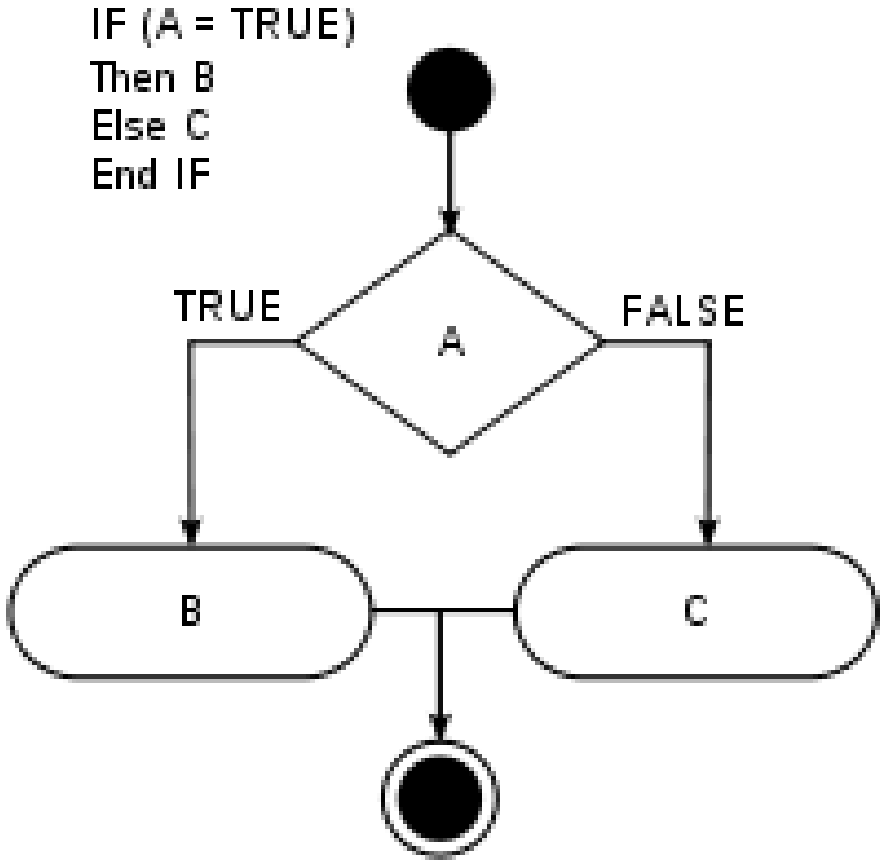
BLOKINĖ SCHEMA

Grafinis algoritmų vaizdavimo būdas, kai atskiros operacijos vaizduojamos įvairiais blokais, o operacijų vykdymo eiliškumas nurodomas ryšiais tarp blokų.



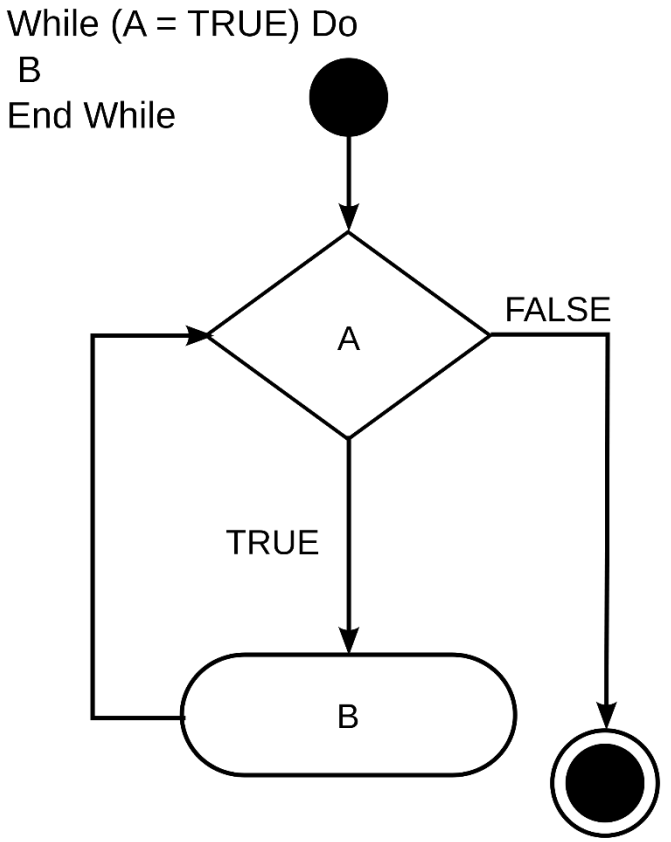
IF BLOKINĖ SCHEMA

```
78  A = True
79  B = True
80  C = True
81  if A:
82      C = False
83  else:
84      B = False
85  print('', A, B, C)
```



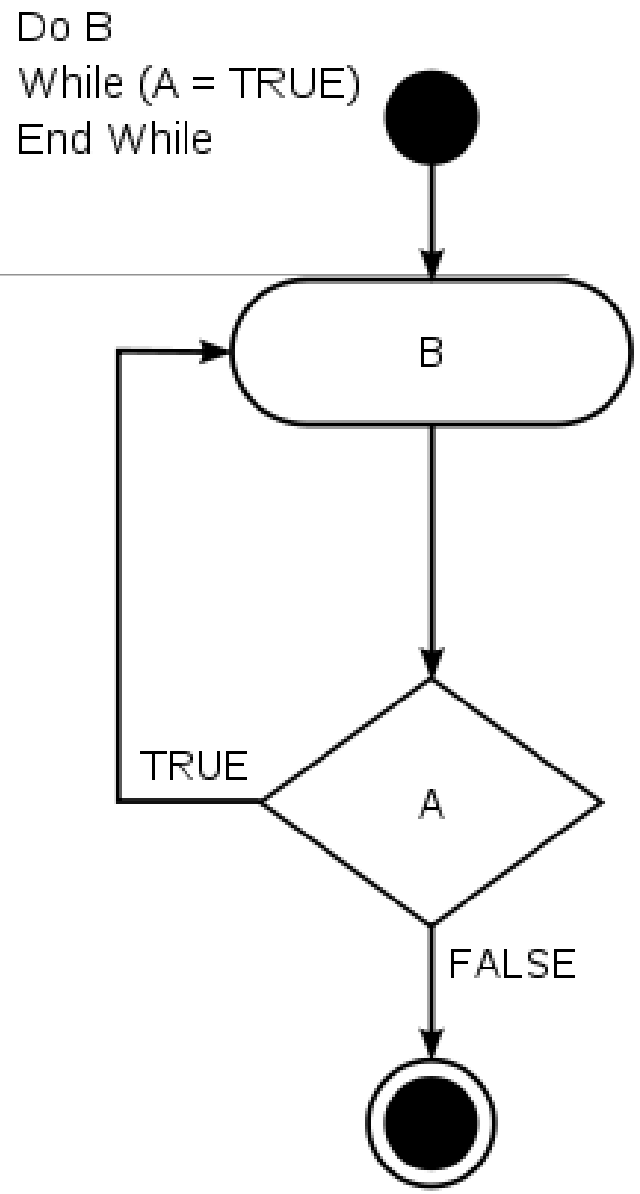
WHILE CIKLO BLOKINĖ SCHEMA

```
3   A = True
4   B = True
5   while A:
6       print(B)
```



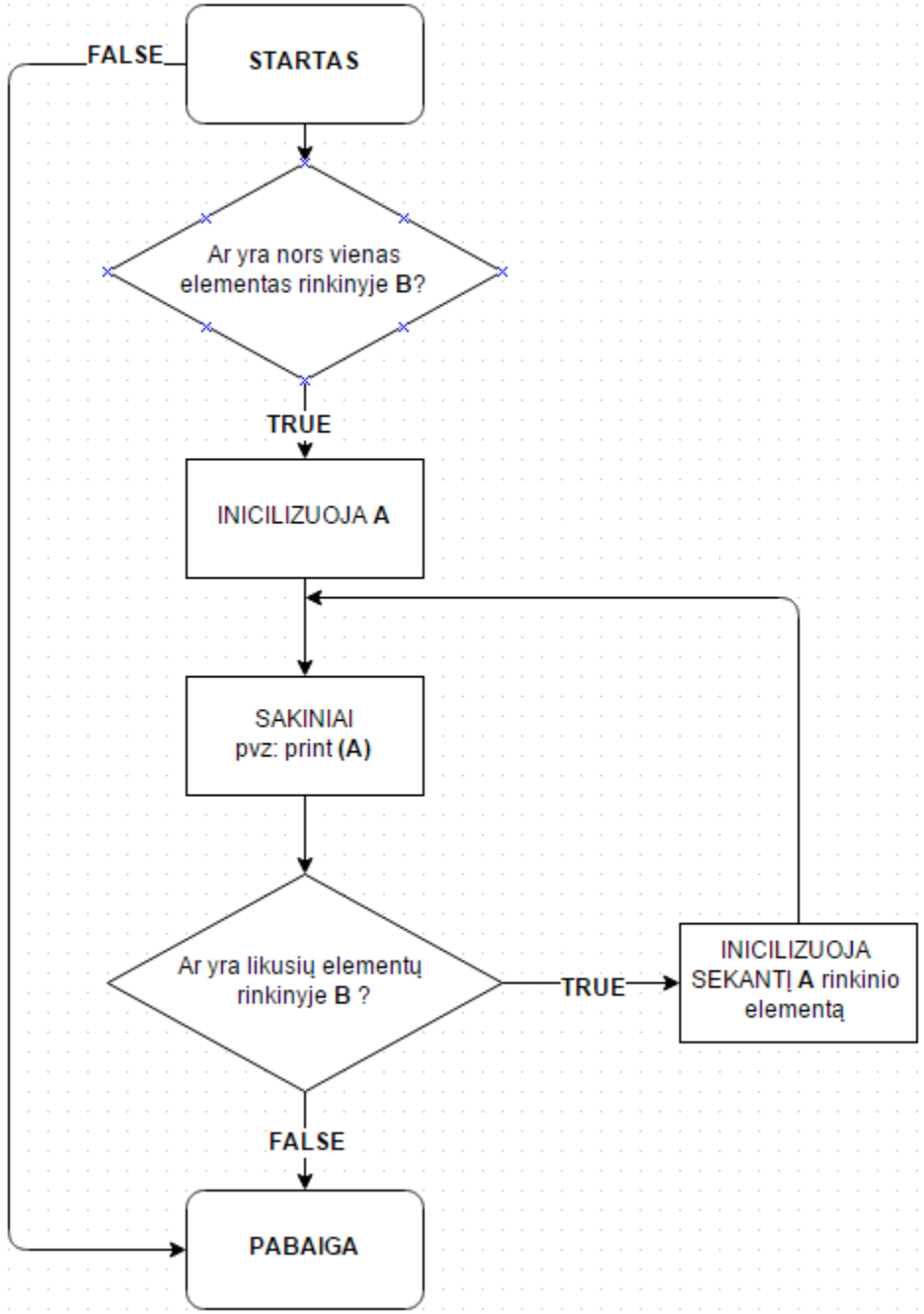
DO WHILE

```
2    A = True
3    B = 0
4    while A:
5        B += 1
6        if B >= 5:
7            break
8        print(B)
```



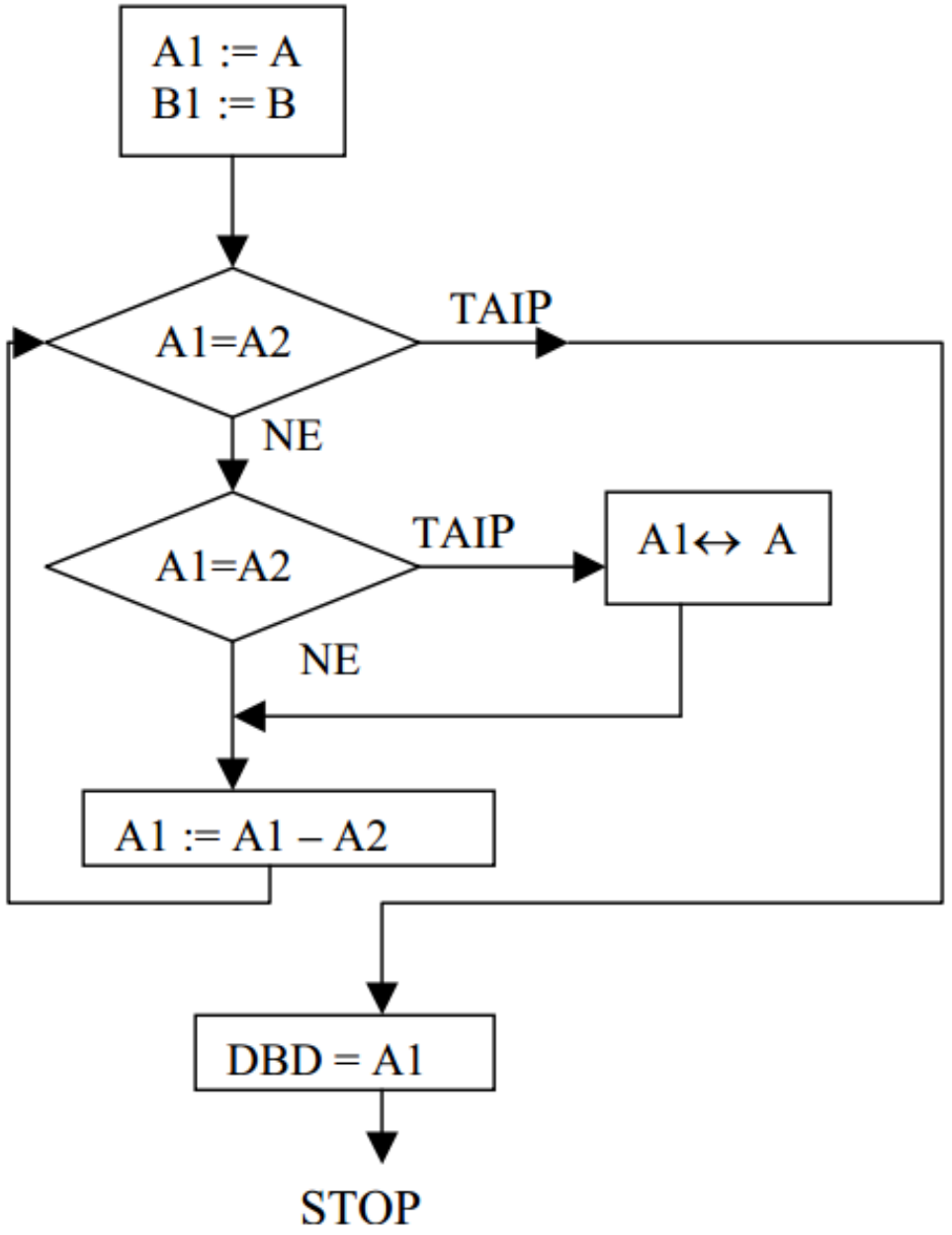
FOREACH CIKLO BLOKINĖ SCHEMA

```
10  B = [10, 20, 30, 40, 50]
11  for A in B:
12      print(A)
13
14  for A in range(0, len(B), 2):
15      print(A)
```



EUKLIDO ALGORITMO BLOKINĖ SCHEMA

```
3   temp = 0
4   A1 = 5000
5   A2 = 3000
6   DBD = 0
7   while A1 != A2:
8       if A1 < A2:
9           temp = A1
10          A1 = A2
11          A2 = temp
12      A1 = A1 - A2
13  DBD = A1
14  print('DBD yra:', DBD)
```

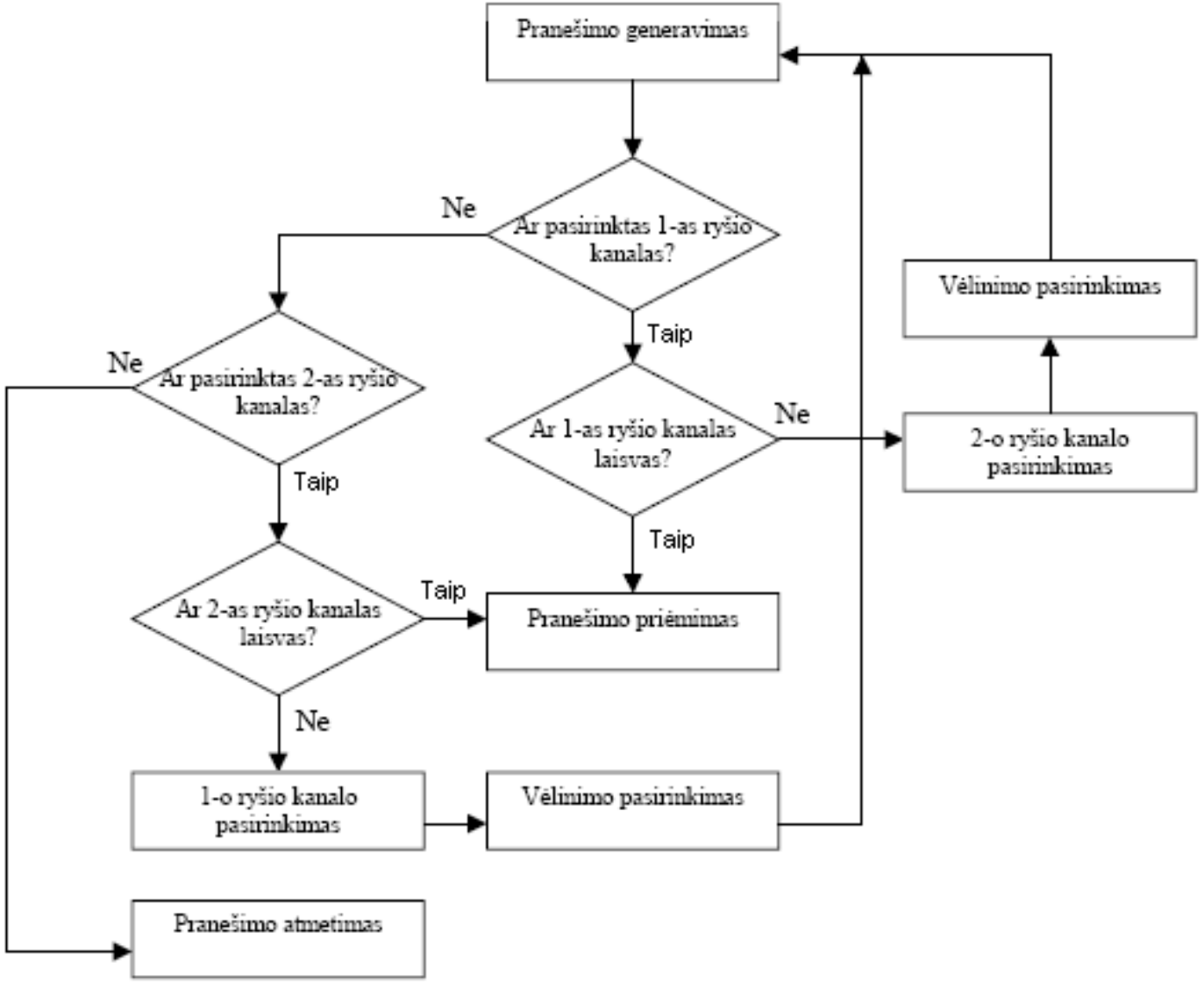


EUKLIDO ALGORITMAS REKURSYVUS

```
16  t = 0
17  def DBD(a, b):
18      print('a =', a, ', b =', b)
19      t = b
20      b = a % b
21      print('b =', b, ', t =', t)
22      if b == 0:
23          print('DBD yra:', t)
24          return t
25      else:
26          return DBD(t, b)
27  DBD(522, 14)
```

VEIKSMAI:

```
a = 522 , b = 14
b = 4 , t = 14
a = 14 , b = 4
b = 2 , t = 4
a = 4 , b = 2
b = 0 , t = 2
DBD yra: 2
```


Pranešimo generavimas
Ar pasirinktas 1-as ryšio kanalas?
Ne
Taip
Ar pasirinktas 2-as ryšio kanalas?
Ne
Taip
Ar 1-as ryšio kanalas laisvas?
Ne
Taip
Vėlinimo pasirinkimas
2-o ryšio kanalo pasirinkimas
Ar 2-as ryšio kanalas laisvas?
Taip
Pranešimo priėmimas
Ne
1-o ryšio kanalo pasirinkimas
Vėlinimo pasirinkimas
Pranešimo atmetimas

PSEUDOKODAS KAS TAI ?

Pseudokodas yra metodika algoritmo užrašymui žmogui suprantama kalba, pavyzdžiui:

- Matematiškai, valstybine šnekamąja kalba, programuotojui suprantama kalba ir panašiai.
- Taisyklės rašant pseudokodą:
- Negali būti per daug išsiplėsta, t.y. rašyti konkrečiai, kad bet kuris skaitovas galėtų suprasti ir realizuoti aprašytąjį algoritmą.

PSEUDOKODAS KAS TAI ?

```
{S - stekas}
Push(S, v)
MarkV(v) {pažymi aplankytą viršūnę}
While (not IsEmpty(S)) {
 If (neegzistuoja neaplankyta kaimynė S viršūnei)
      then Pop(S)
      else {
            imame u - S viršūnės kaimynę (u ∈ V)
            Push(S, u)
            MarkV(u)
      }
}
```

PSEUDOKODAS KAS TAI ?

```
{Q - eilė}
Add(Q, V) {įdedame viršūnę V į eilę}
MarkV(V) {pažymi aplankytą viršūnę}
While (not IsEmpty(Q)) do {
      w:= Get(Q)
      for ∀ w neaplankytai kaimynei u do {
            MarkV(u)
            Add(Q, u)
      }
}
```

MATEMATIŠKAI PSEUDOKODAS

INPUT: $\psi_1, \ldots, \psi_m, v_{T,min}, V_{T,max}, \varepsilon$
OUTPUT: Φ, Ψ
INITIALIZATION
$\Phi \leftarrow \{v_{T,min}\}$/*$\Phi$ contains the equality points of functions in $\{\psi_1, \ldots, \psi_m\}$*/
$\Psi \leftarrow \{\emptyset\}$/*$\Psi$ contains the modes of operation ℓ that maximize option value*/
FOR $\ell = 1 : (m-1)$
  Find points $v_{T,i}$ where $\psi_\ell(v_T) - \psi_j(v_T) = 0$,
    for all $j = \ell + 1, \ldots m$,
    within error bound ε using a root finding algorithm.
  $\Phi \leftarrow \Phi \cup \{v_{T,i}\}$
END FOR
  $\Phi \leftarrow \Phi \cup \{v_{T,max}\}$
  $\Phi \leftarrow$ Rank(Φ)/* Re-order Φ in ascending sequence*/
FOR $n = 1 : (|\Phi| - 1)$
$\ell^* = 1$
$max_n = \psi_1(v_{T,n} + \varepsilon)$
  FOR $\ell = 2 : m$
    IF $\psi_\ell(v_{T,n} + \varepsilon) > max_n$
      $max_n = \psi_\ell(v_{T,n} + \varepsilon)$
      $\ell^* = \ell$
    END IF
  END FOR
  $\Psi \leftarrow \Psi \cup \{\ell^*\}$
END FOR

EUKLIDO ALGORITMO PSEUDOKODAS

Algorithm 1 Euclids algorithm

```
1: procedure EUCLID(a, b)                    ▷ The g.c.d. of a and b
2:     r ← a mod b and foobar
3:     while r ≠ 0 do                        ▷ We have the answer if r is 0
4:         a ← b
5:         b ← r
6:         r ← a mod b
7:     end while
8:     return b                              ▷ The gcd is b
9: end procedure
```

In Algorithm 1, variable `foobar` (in line 2), corresponds to...

EUKLIDO ALGORITMO REALIZACIJA Python

```python
__author__ = 'Marius'

# Didžiausio bendrojo daliklio algoritmas

def DBD(A, B): # A ir B, DBD skaičiavimui
    while B != 0: # Kol B nelygus nuliui ieškom
        temp = B  # B patalpinama į laikinąjį kintamąjį
        B = A % B # A mod B, jei = nulis tai yra DBD
        A = temp # A priskiriame laikinąjį kintamąjį
    return A # Gražinamas DBD kadangi buvo išsaugotas temp
print(DBD(950, 100))
```

Realizacijos programų išeities tekstai (angl. source code) yra talpinami moodle!

EUKLIDO ALGORITMO REALIZACIJA

Pradinės reikšmės: A = 950 B = 100

1 ciklo iteracija. Į laikiną „temp“ kintamąjį patalpinama B = 100 , tada 950 **mod** 100 = 50, liekana yra lygi 50, todėl yra tęsiamas algoritmas.

2 ciklo iteracija. Į laikiną „temp“ kintamąjį patalpinama B = 50 , tada 100 **mod** 50 = 0, liekana yra 0 algoritmas sustoja ir gražina 50 kaip didžiausią bendrąjį daliklį.

Didžiausias bendras daliklis **(DBD)** yra**: 50**

KLAUSIMAI ?