



Algoritmų parinkimas.

Sąlyginiai sakiniai (If)

Algoritmo savoka

Kas yra algoritmas?

- Algoritmas – tai žingsnis po žingsnio instrukcijos, skirtos problemai išspręsti.

Kodėl svarbu parinkti tinkamą algoritmą?

- Tinkamas algoritmo pasirinkimas leidžia efektyviau spręsti problemas ir sutaupyti resursus.

Pavyzdys iš gyvenimo:

- Įsivaizduokite, kad turite keliauti į vietą. Galite rinktis trumpiausią maršrutą, bet jis gali būti užsikimšęs eismu. Panašiai kaip ir algoritmuose, kartais greičiausias būdas ne visada yra pats efektyviausias.

Sąlyginiai sakiniai (If)

Kas yra sąlyginiai sakiniai?

- Sąlyginiai sakiniai leidžia programai priimti sprendimus pagal tam tikras sąlygas.

Kaip tai veikia?

- Programa tikrina sąlygą (pvz., ar skaičius didesnis už 5).
- Jei sąlyga teisinga (True), atliekami tam tikri veiksmai.
- Jei sąlyga neteisinga (False), galima vykdyti alternatyvų veiksmą arba nieko nedaryti.

Pavyzdys:

- „Jei lauke lyja, pasiimk skėtį. Jei ne, gali eiti be jo.“

Pavyzdys

```
1 # Python
2
3 x = 10
4 if x > 5:
5     print("x yra didesnis už 5")
6     # operacija if sąlygoje
7 # if sąlygos pabaiga
```

```
1 // C++
2
3 int x = 10;
4 if (x > 5) {
5     cout << "x yra didesnis už 5" << endl;
6     // operacija if sąlygoje
7 }
8 // if sąlygos pabaiga
```

Python ir C++ sintaksės skirtumai

■ Python:

- Naudoja **įtraukas** (indentation), kad parodytų kodo blokų ribas.
- Kiekvienas sąlyginio sakinio blokas turi būti tinkamai įtrauktas.

■ C++:

- Naudoja **riestinius skliaustus** `{ }` kodo blokams nurodyti.
- Kodo blokai gali būti parašyti be papildomų įtraukų (nors rekomenduojama naudoti įtraukas siekiant aiškumo).

■ Esminis skirtumas:

- **Python** pasikliauja įtraukomis kaip būtina sintaksės dalimi.
- **C++** naudojami riestiniai skliaustai `{ }`, kurie leidžia lankstumą, bet padidina klaidų tikimybę, jei nėra tinkamai naudojamos įtraukos

(If) Logika ir naudojimas

■ Sąlyginio sakinio logika:

- **Tikrina sąlygą:** Arba **teisinga** (True), arba **klaidinga** (False).
- **Veiksmai:** Jei sąlyga teisinga, atliekami nurodyti veiksmai. Jei klaidinga, atliekami alternatyvūs veiksmai arba nieko nevyksta.

■ Kada naudoti sąlyginius sakinius?

- **Sprendimų priėmimui** programoje.
- **Skirtingų scenarijų valdymui:**
 - Pvz.: Patikrinimas, ar vartotojo įvestis atitinka tam tikras sąlygas.
 - Pvz.: Kontrolė programos srauto pagal situacijas (pvz., ar pakanka resursų).

■ Pavyzdys:

- „Jei yra pakankamai vietos atmintyje, tęsti programos vykdymą. Jei ne, parodyti klaidos pranešimą.“

Algoritmų parinkimas naudojant „If“

- Užduotis: Patikrinkite, ar skaičius yra **teigiamas**, **nulis**, ar **neigiamas**.

```
1 | # Python
2 |
3 | num = int(input("Įveskite skaičių: "))
4 | if num > 0:
5 |     print("Teigiamas")
6 | elif num == 0:
7 |     print("Nulis")
8 | else:
9 |     print("Neigiamas")
```

```
1 | // C++
2 |
3 | int num;
4 | cout << "Įveskite skaičių: ";
5 | cin >> num;
6 | if (num > 0) {
7 |     cout << "Teigiamas" << endl;
8 | }
9 | else if (num == 0) {
10 |     cout << "Nulis" << endl;
11 | }
12 | else {
13 |     cout << "Neigiamas" << endl;
14 | }
```

Kaip naudojant „If“ galima pasirinkti skirtingus veiksmus pagal įvesties sąlygas

- **Kiekviena sąlyga lemia skirtingą veiksmą:**
 - **Sąlygos tikrinamos iš eilės:** Programa tikrina, ar sąlyga teisinga, ir atlieka veiksmus, kai randa pirmą teisingą sąlygą.
- **Pavyzdys:**
 - Jei **num > 0**, programa atspausdina „Teigiamas“.
 - Jei **num == 0**, programa atspausdina „Nulis“.
 - Jei **num < 0**, programa atspausdina „Neigiamas“.
- **Kaip tai veikia?**
 - Sąlyginiai sakiniai leidžia algoritmui prisitaikyti prie skirtingų situacijų, vykdant skirtingus veiksmus, priklausomai nuo įvesties duomenų.

Santrauka

- Algoritmo savoka
- Sąlyginiai sakiniai (If)
- Python ir C++ sintaksės skirtumai
- (If) Logika ir naudojimas
- Algoritmų parinkimas naudojant „If“