Algoritmai ir programavimas

Rekursija

Tikslai

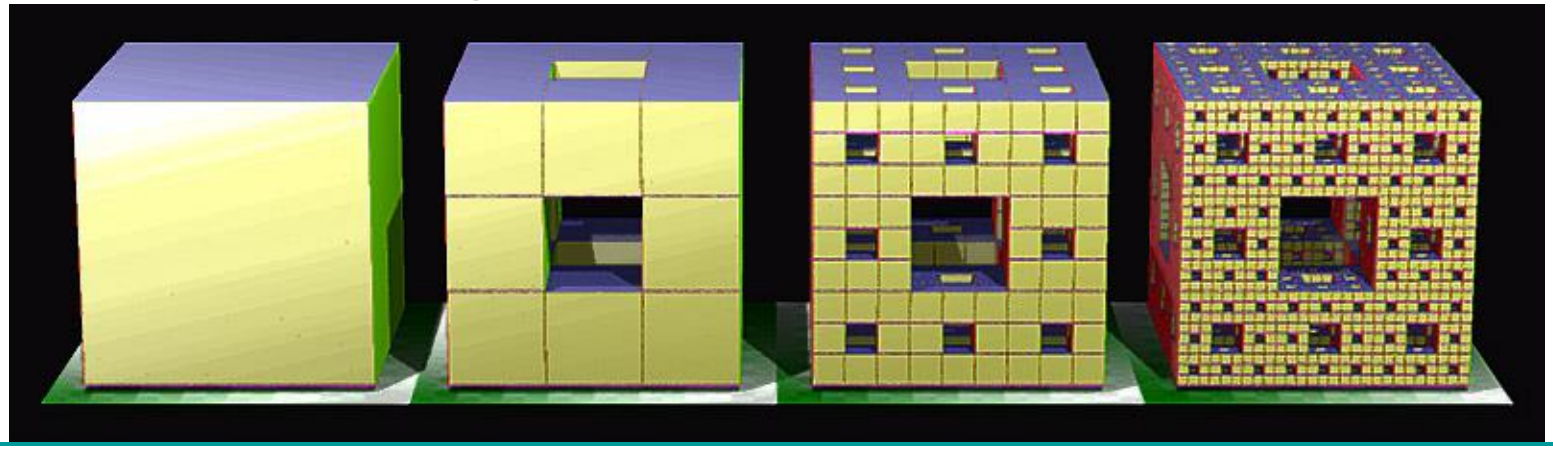
- Suprasti rekursijos sąvoką ir principus
- Išmokti rašyti rekursines funkcijas programavimo kalboje
- Suvokti rekursijos taikymo sritis ir privalumus/trūkumus

Kas yra rekursija

Rekursija – tai procesas, kai funkcija kviečia pati save spręsti mažesnę problemos versiją

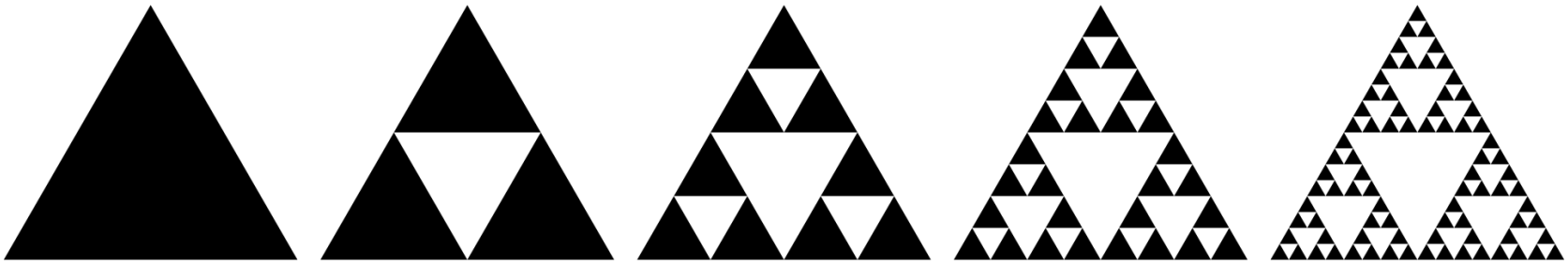
Mengerio kempane (geometrinė figūra)

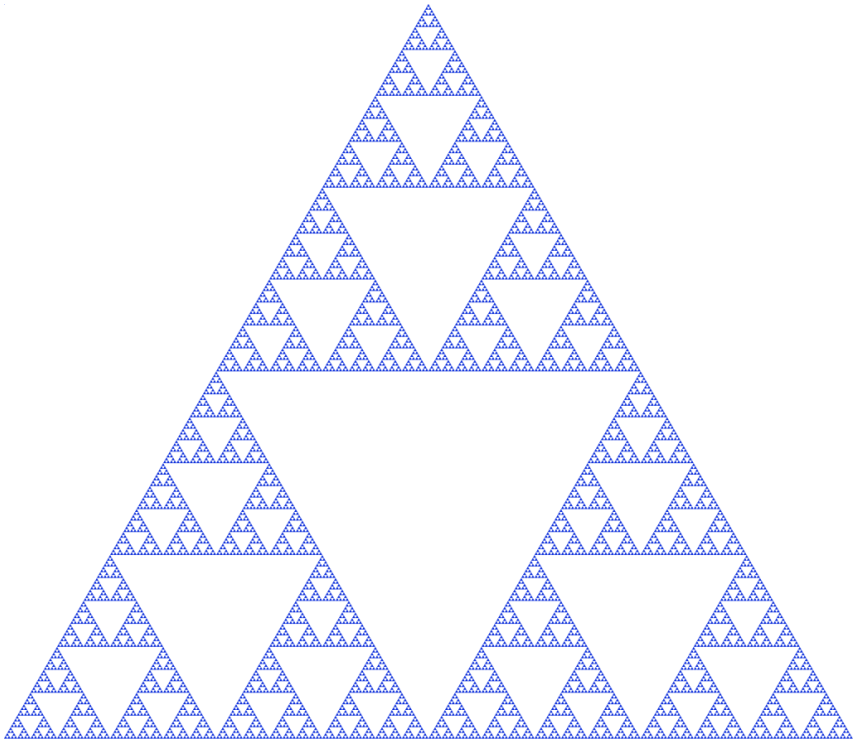
- Konstravimo algoritmas:
  - Paimamas kubas.
  - Kubas suskaidomas į 27 vienodo dydžio kubelius.
  - Pašalinamas kubo viduryje esantis kubelis, taip pat dar 6 kiekvienos sienos viduryje esantys kubeliai.
  - Toliau su kiekvienu likusiu kubeliu veiksmai kartojami nuo antro žingsnio



** Šaltinis: https://inf-knyga.nmakademija.lt/lt/latest/04_rekursija.html#f12*

Sierpinskio trikampis





* Šaltinis: https://en.wikipedia.org/wiki/Sierpi%C5%84ski_triangle

Pavyzdys iš realaus gyvenimo

„Veidrodžių kambarys“ – stovint tarp dviejų veidrodžių matome begalinį vaizdo atspindį, panašiai veikia ir rekursija

* Šaltinis: https://taylordaug.github.io/blog/tech.html

Rekursijos pavyzdys (Faktorialo skaičiavimas)

```python
def faktorialas(n):
    if n == 0:
        return 1
    else:
        return n * faktorialas(n-1)
```

Python

```cpp
long long fakt(int n) {
    if(n == 0) return 1;
    return n*fakt(n-1);
}
```

C++

Rekursijos terminai

- **Bazinis atvejis**: momentas, kai rekursija sustoja (pvz., kai n == 0).
- **Rekursinis atvejis**: funkcija kviečia pati save spręsdama mažesnę problemos versiją

Praktika

- Apskaičiuoti Fibonačio seką

Fibonacci seka (Fibonãčio sekà), *skaičių seka, kurios kiekvienas paskesnis narys lygus dviejų prieš jį esančių narių sumai, t. y. Fi+2 = Fi+1 + Fi, i = 1, 2, …; F1 = F2 = 1. Pirmieji sekos nariai yra 1, 1, 2, 3, 5, 8, 13, 21, ….*

** Šaltinis: https://www.vle.lt/straipsnis/fibonacci-seka/#:~:text=Fibonacci%20seka%20(Fibon%C3%A3%C4%8Dio%20sek%C3%A0)%2C, sekos%20nariai%20vadinami%20Fibonacci%20skai%C4%8Diais.*

Fibonačio seka

```python
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```

Python

```cpp
long long F(int n) {
    if(n <= 2) return 1;
    return F(n-1) + F(n-2);
}
```

C++

Kur naudojama rekursija?

- Algoritmai: binarės paieškos medis, rūšiavimas, tokie kaip greitojo rūšiavimo algoritmas.
- Matematinės problemos: skaičių sekos, kombinatorika.
- Žaidimų kūrimas: algoritmai, sprendžiantys galvosūkius, maršrutų paieška

Rekursijos privalumai

- Lengvesnis problemos supratimas ir sprendimas
- Švaresnis ir elegantiškesnis kodas
- Natūralus medžių struktūrų sprendimas
- Tinka tam tikrų algoritmų įgyvendinimui
- Intuityvus sprendimų modeliavimas
- Naudojimas, kai tiksli iteracija nežinoma

Rekursijos trūkumai

- Didelis atminties naudojimas
- Lėtesnis veikimas
- Sunkesnis supratimas ir derinimas
- Rizika įstrigti begalinėje rekursijoje
- Neefektyvumas tam tikromis sąlygomis
- Platformos apribojimai

Klausimai ir diskusijos

- Kur dar galite pritaikyti rekursiją savo gyvenime ar kituose mokomuosiuose dalykuose?

Užduotis namuose

- Sugalvoti problemą, kurią galima būtų išspręsti naudojant rekursiją, ir parašyti funkciją

Daugiau

- https://inf-knyga.nmakademija.lt/lt/latest/04_rekursija.html#f12
- https://algol.dev/en/recursion-what-is-it/
- https://en.wikipedia.org/wiki/Recursion_(computer_science)
- https://blog.skillfactory.ru/glossary/rekursiya/
- https://practicum.yandex.ru/blog/rekursiya-v-programmirovanii/

Pabaiga