
Duomenų struktūrų naudojimas.

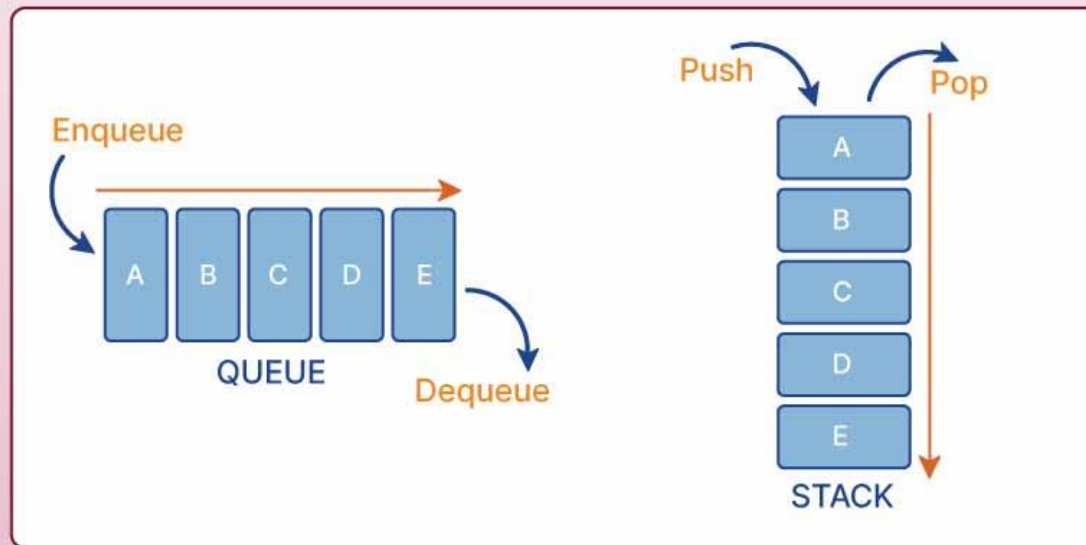
Praėjusios pamokos santrauka

- FIFO, LIFO sąvokos
- stekai / deklai (dekas)
- jų skirtumai
- pagrindinės operacijos su steku / deku
- Python ir C++ implementacijų skirtumai

FIFO (First In, First Out)

LIFO (Last In, First Out)

Difference Between FIFO and LIFO



Duomenų struktūros

- Suprasti pagrindines duomenų struktūrų sąvokas
 - masyvai
 - susietieji sąrašai
 - stekai / deklai (dekas)
 - **eilės**
- Įgyvendinimas naudojant Python ir C++
- Suprasti įvairių duomenų struktūrų privalumus ir trūkumus

Ką išmoksime šiandien

- Tyrinėsime, kaip veikia eilės
- Lyginsime stekų, deklų ir eilių funkcionalumą
- Darysime praktines užduotis, kad geriau suprastumėte jų skirtumus ir kaip juos pritaikyti sprendžiant įvairias problemas

Kas yra eilė?

- Eilė – tai **pirmo įėjimo, pirmo išėjimo (FIFO)** duomenų struktūra.
- Elementai pridedami į galą ir pašalinami iš priekio



Panašu į laukimo eilę, kur pirmas žmogus eilėje yra aptarnaujamas pirmas.

Dažnos operacijos

- **enqueue**: Pridėjimas į eilės galą.
- **dequeue**: Pašalinimas iš eilės priekio.
- **peek**: Peržiūrėti pirmąjį elementą eilėje, jo neištrinant.

Python ir C++ įgyvendinimas

- Python: Naudojame `deque` modulį eilės įgyvendinimui.
- C++: Naudojame `std::queue` klasę pagrindinėms eilės operacijoms.

Python – Eilės naudojant „deque“

```
1 from collections import deque
2
3 # Sukuriame eilę
4 queue = deque()
5
6 # Pridedame elementus (enqueue)
7 queue.append("Element 1")
8 queue.append("Element 2")
9
10 # Pašaliname elementą (dequeue)
11 first = queue.popleft()
12
13 # Peržiūrimė pirmą elementą
14 peek_element = queue[0]
15
16 print(f"Pirmas elementas: {peek_element}")
```

C++ – Eilės naudojant „std::queue“

```
1  #include <iostream>
2  #include <queue>
3
4  int main() {
5      std::queue<std::string> queue;
6
7      // Pridedame elementus (enqueue)
8      queue.push("Element 1");
9      queue.push("Element 2");
10
11     // Pašaliname elementą (dequeue)
12     queue.pop();
13
14     // Peržiūrimė pirmą elementą
15     std::cout << "Pirmas elementas: " << queue.front() << std::endl;
16
17     return 0;
18 }
```

Svarbiausios operacijos

- **Python:** `append()` ir `popleft()`.
- **C++:** `push()` ir `pop()`.

Praktika

- Simuliuokite banko operacijų valdymo sistemą, naudodami eilę, steką ir deklą, kad pamatytumėte jų skirtumus atliekant tą pačią užduotį.
- **Scenarijus:**
 - Bankas turi apdoroti klientų transakcijas. Kiekvienas klientas gali atvykti, atlikti transakciją, arba atšaukti ją. Klientai turi būti aptarnaujami skirtingais prioritetais, priklausomai nuo pasirinktos duomenų struktūros

Scenarijus

Eilė:

- Klientai aptarnaujami atvykimo tvarka (FIFO).
- Naudojamas tais atvejais, kai klientai aptarnaujami pagal atvykimo laiką, kaip banko kasose.

Stekas:

- Paskutinis atvykęs klientas aptarnaujamas pirmas (LIFO).
- Naudojamas situacijose, kai prioritetas skiriamas paskutinei įvykusiai transakcijai, pvz., kai skubūs pavedimai yra svarbesni.

Deklas:

- Klientai gali būti aptarnaujami ir iš priekio, ir iš galo, atsižvelgiant į poreikius.
- Naudojamas, kai kartais pirmi klientai turi prioritetą, o kartais reikia aptarnauti paskutinius, pvz., VIP klientai gali būti praleidžiami į eilės priekį.

Užduotis

- Sukurkite banko operacijų valdymo simuliaciją, kur kiekvienas klientas gali atlikti vieną iš šių veiksmų:
 - Atlikti transakciją (pridėti į eilę/steaką/deklą).
 - Atšaukti transakciją (pašalinti iš eilės/steako/deklo).
 - Peržiūrėti pirmąjį/paskutinį klientą (priklausomai nuo struktūros).
- Simuliacija Python ir C++:
 - Naudokite eilę, steką ir deklą, kad pamatytumėte, kaip kiekviena struktūra keičia aptarnavimo tvarką ir prioritetus.
 - Įgyvendinkite tas pačias funkcijas visose trijose struktūrose, kad stebėtumėte skirtumus.

Tikslas:

- Palyginti, kaip kiekviena duomenų struktūra (eilė, stekas, deklas) keičia klientų aptarnavimo tvarką ir kaip jų naudojimas gali būti pritaikytas realiose situacijose.

Refleksija

- Pagalvokite kaip skirtingos duomenų struktūros keičia aptarnavimo tvarką ir elgesį su duomenimis (FIFO, LIFO, ir dvipusės operacijos su deklų)



Pabaiga