



>>>

Web Development with Python Lesson 11





OBSAH PREZENTÁCIE

- Opakovanie dátových štruktúr - list, tuples, sets, dictionary
- Linked List
- Stack
- Queue
- Binary tree

OPAKOVANIE

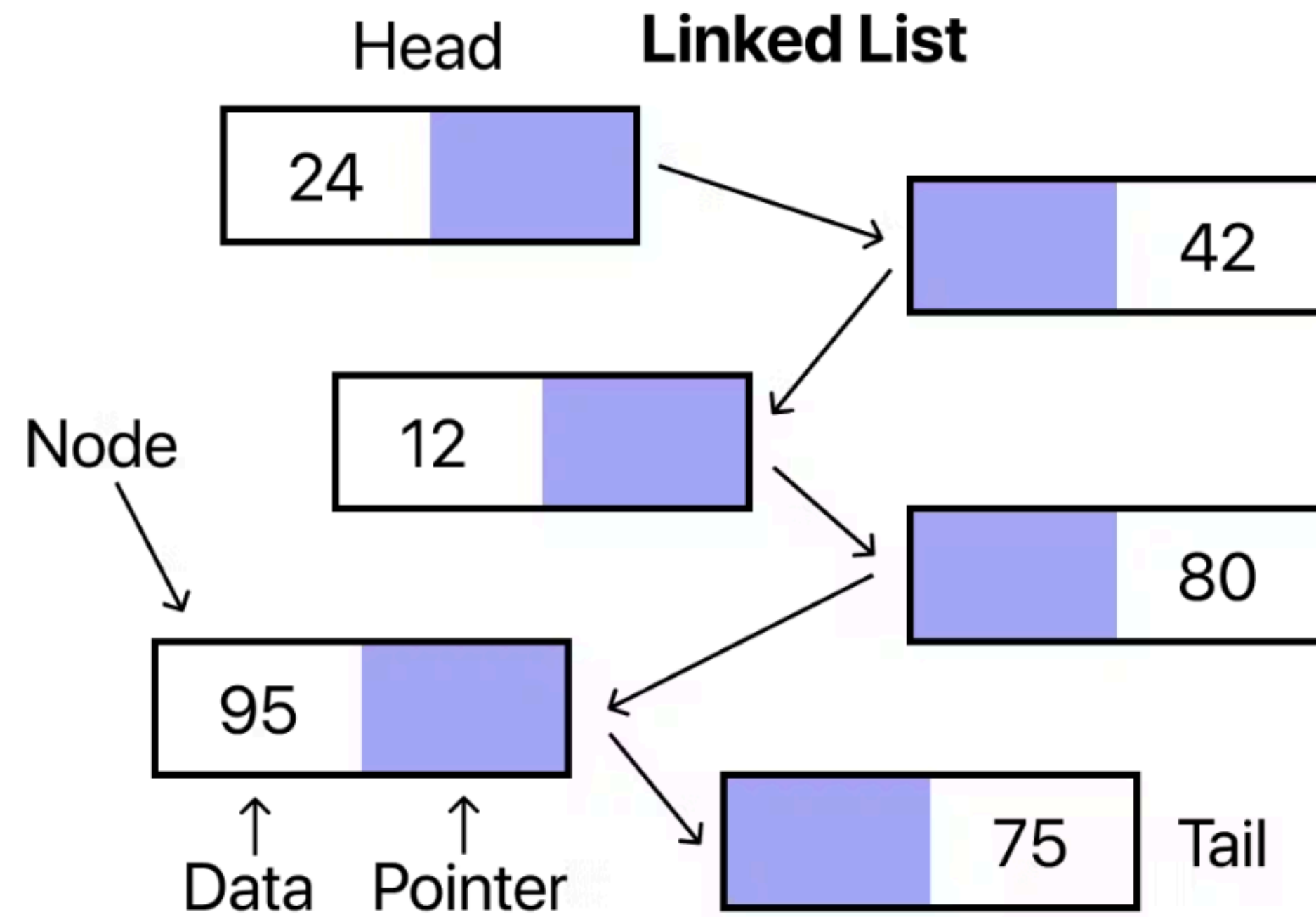
- Ako definujete OOP?
- 4 piliere OOP
- Vysvetlite každý pilier
- Čo je statické metóda
- Čo robí magická funkcia `__str__`
- Čo robí magická funkcia `__eq__`
- Môže sa viacnásobne dediť v Pythone?

LIST, TUPLES, SETS, **DICT**

- **list** - zoznamy sú usporiadané kolekcie, ktoré môžu obsahovať prvky rôznych typov. Sú indexované a ich prvky môžu byť zmenené
- **set** - neusporiadané kolekcie bez duplicitných prvkov
- **tuple** - n-tice sú usporiadané a nemenné kolekcie, čo znamená, že raz vytvorené, ich prvky už nie je možné zmeniť
- **dict** - kolekcie kľúč-hodnota, kde každý kľúč je unikátny

LINKED LIST

Array		
0	24	0x100
1	42	0x104
2	12	0x108
3	80	0x112
4	95	0x116
5	75	0x120
↑ Index		↑ Memory Locations



LINKED LIST

	Array	Linked List
Cost of accessing elements	$O(1)$	$O(n)$
Insert/Remove from beginning	$O(n)$	$O(1)$
Insert/Remove from end	$O(1)$	$O(n)$
Insert/Remove from middle	$O(n)$	$O(n)$

POUŽITIE LINKED LIST

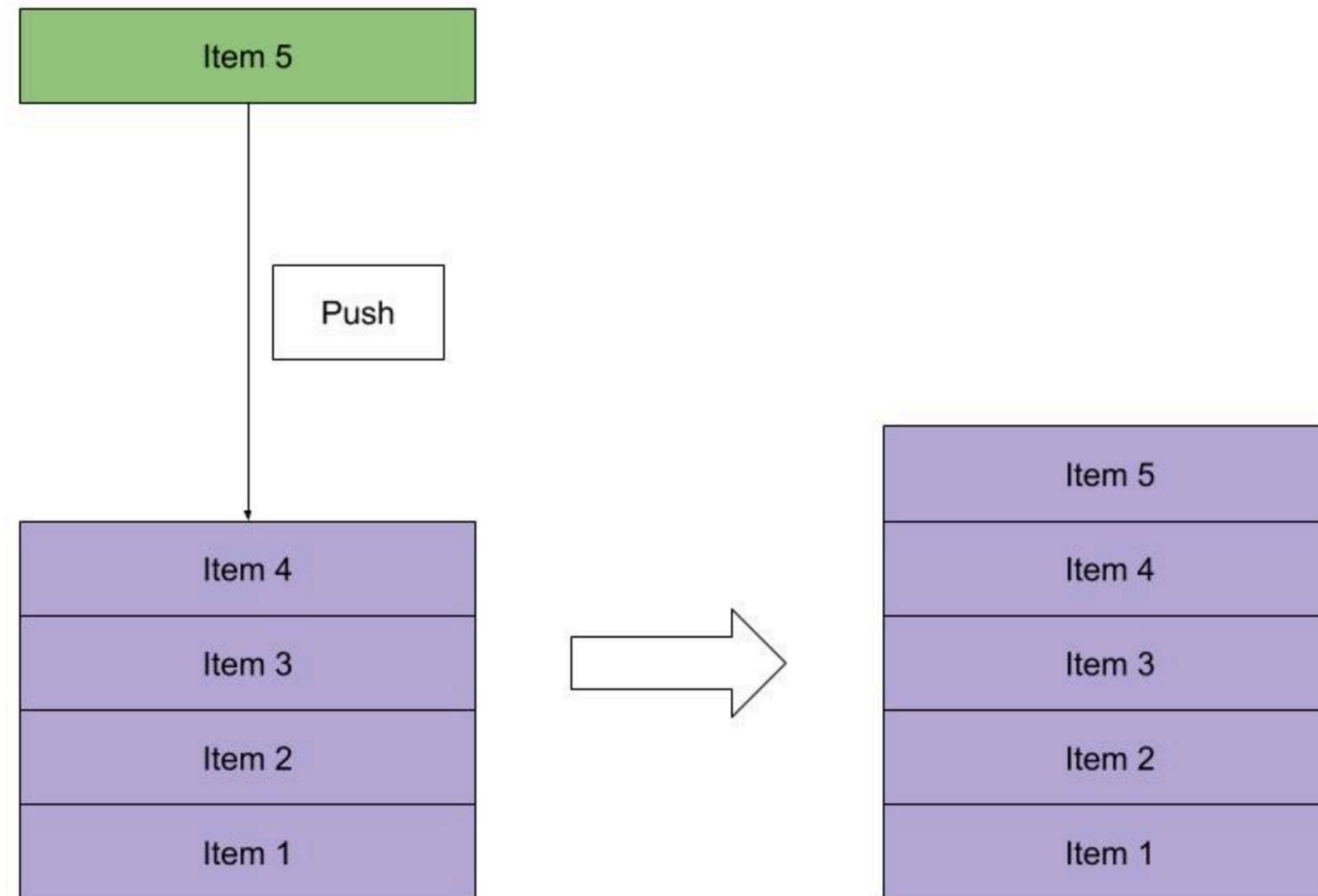
- v hudobných prehrávačoch môže byť spájaný zoznam použitý na implementáciu playlistu, kde používatelia môžu ľahko pridávať, odstraňovať a preskakovať skladby
- operačné systémy môžu používať spájané zoznamy na správu pamäte, najmä pri alokácii a dealokácii blokov pamäte. V dôsledku flexibilnej povahy spájaných zoznamov je jednoduchšie spravovať rôzne veľkosti pamäťových blokov.

IMPLEMENTÁCIA LINKED LIST

Naprogramujte Linked list v Pythone. Vytvorte triedu LinkedList, ktorá bude mať atribút “head” kde bude uložený prvý prvok v zozname. Ďalšia trieda je Prvok, ktorá ma dva atribúty “data” a “dalsi_prvok”. Vytvorte funkciu “vloz” v triede “LinkedList” ktora prida novy prvok.

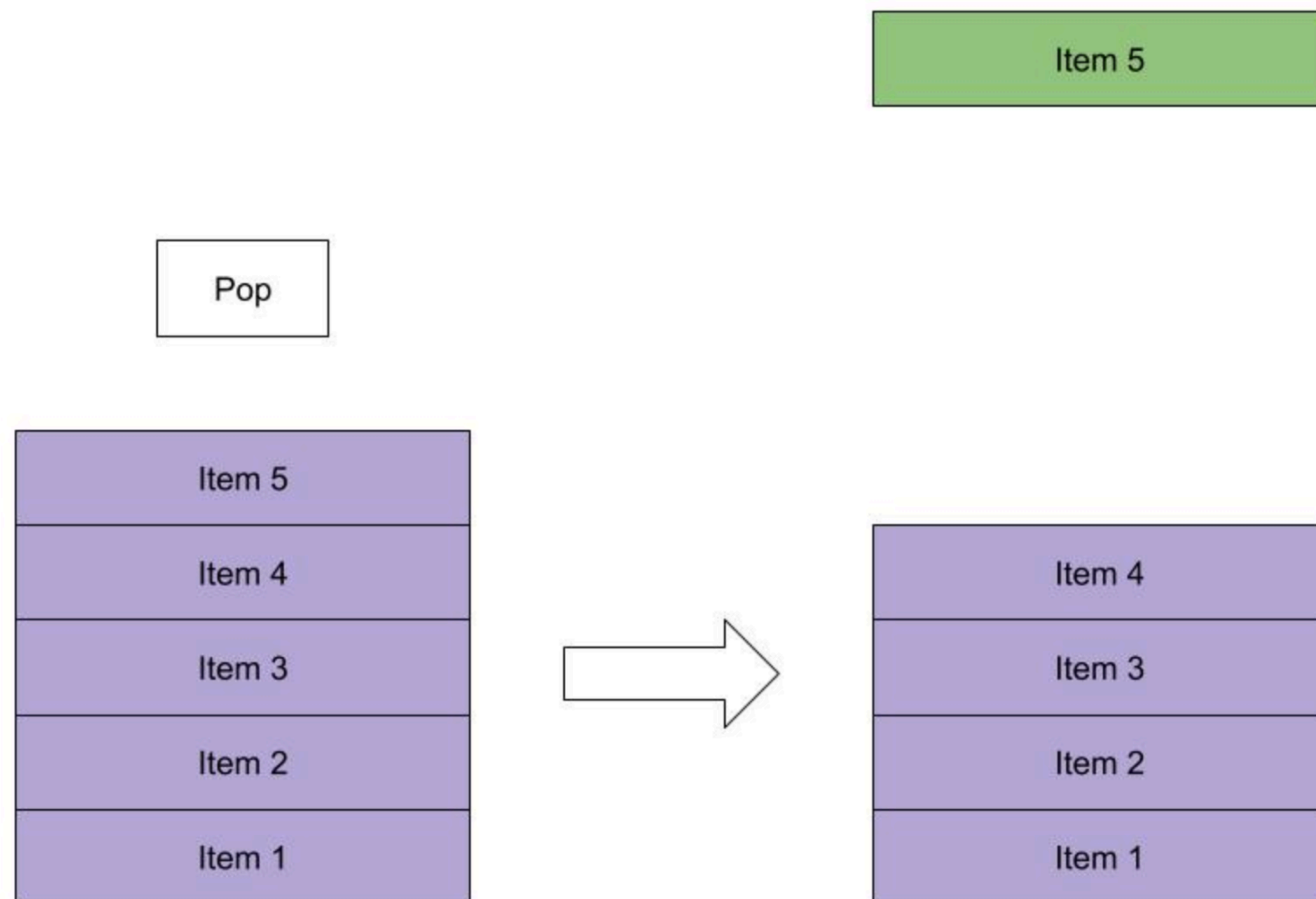
STACK

dátová štruktúra, ktorá sa riadi **Last-in-First-Out** (LIFO) princípom



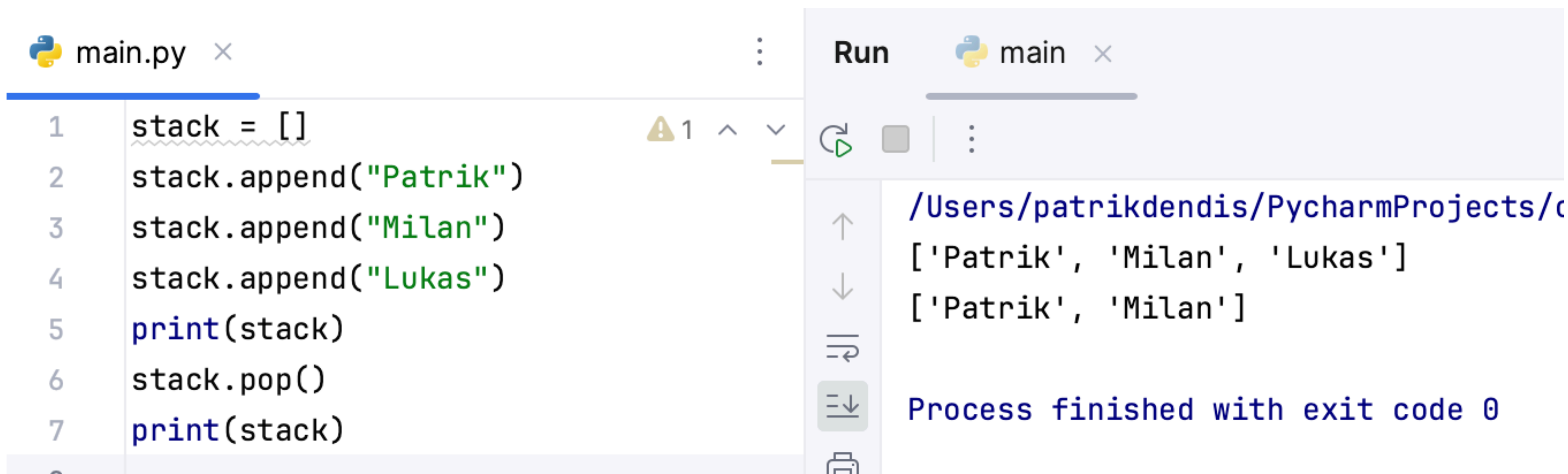
STACK

dátová štruktúra, ktorá sa riadi **Last-in-First-Out** (LIFO) princípom



STACK

môžete použiť list na simuláciu zásobníka použitím metód **append()** na pridanie prvku na koniec zoznamu a **pop()** na odstránenie posledného prvku zoznamu.



The screenshot shows a Python IDE with a file named `main.py` and a `Run` console. The code in `main.py` simulates a stack using a list:

```
1 stack = []
2 stack.append("Patrik")
3 stack.append("Milan")
4 stack.append("Lukas")
5 print(stack)
6 stack.pop()
7 print(stack)
```

The `Run` console shows the output of the program:

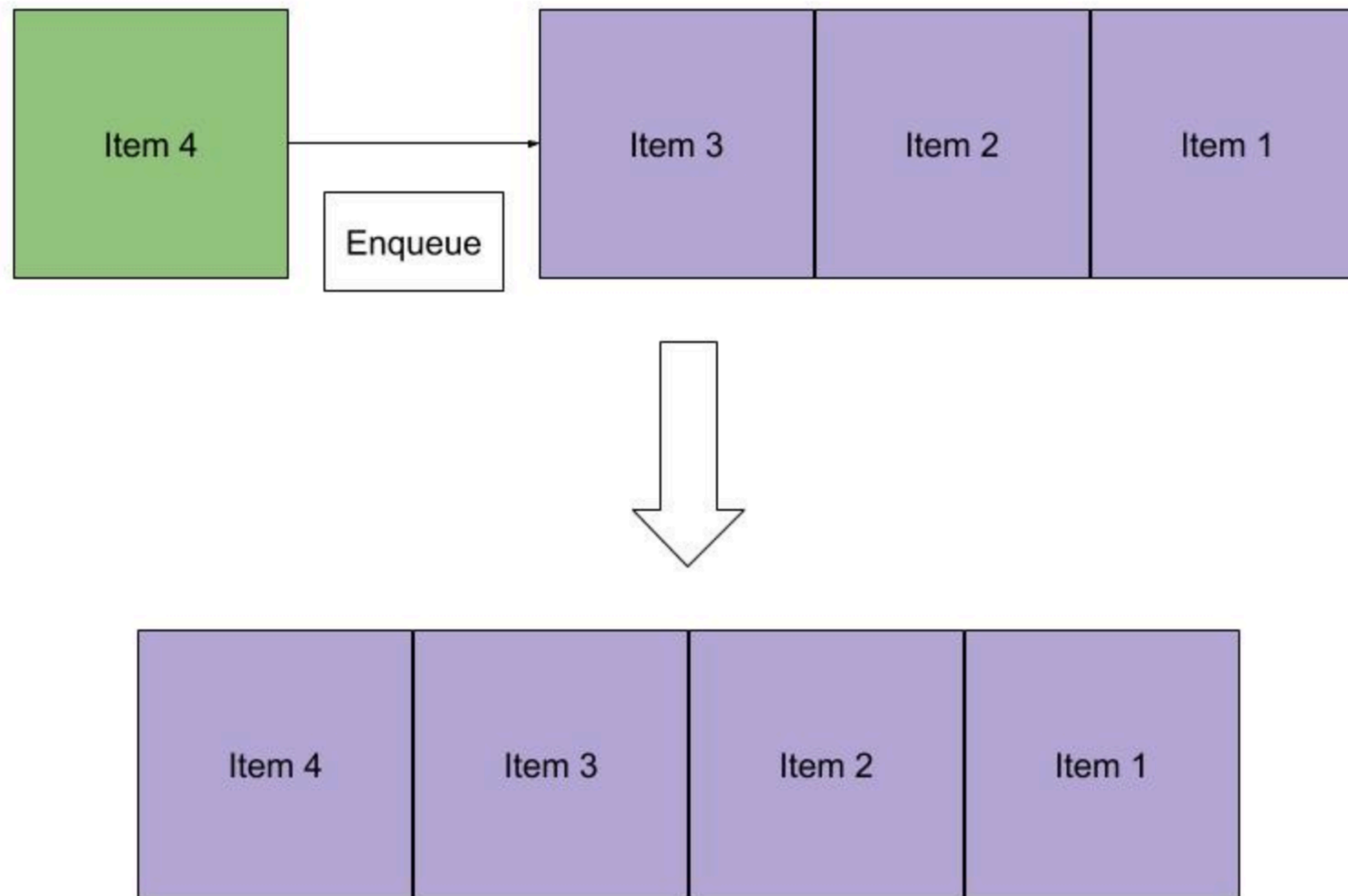
```
/Users/patrikdendis/PycharmProjects/c
['Patrik', 'Milan', 'Lukas']
['Patrik', 'Milan']
Process finished with exit code 0
```

POUŽITIE STACK

- história prehliadača: Web prehliadače často používajú zásobník na ukladanie histórie navštívených stránok, umožňujúc užívateľom vrátiť sa späť na poslednú otvorenú stránku
- v operačných systémoch a programovacích jazykoch sa stack používa na ukladanie informácií o aktívnych funkčných volaniach (tzv. volací zásobník alebo call stack)

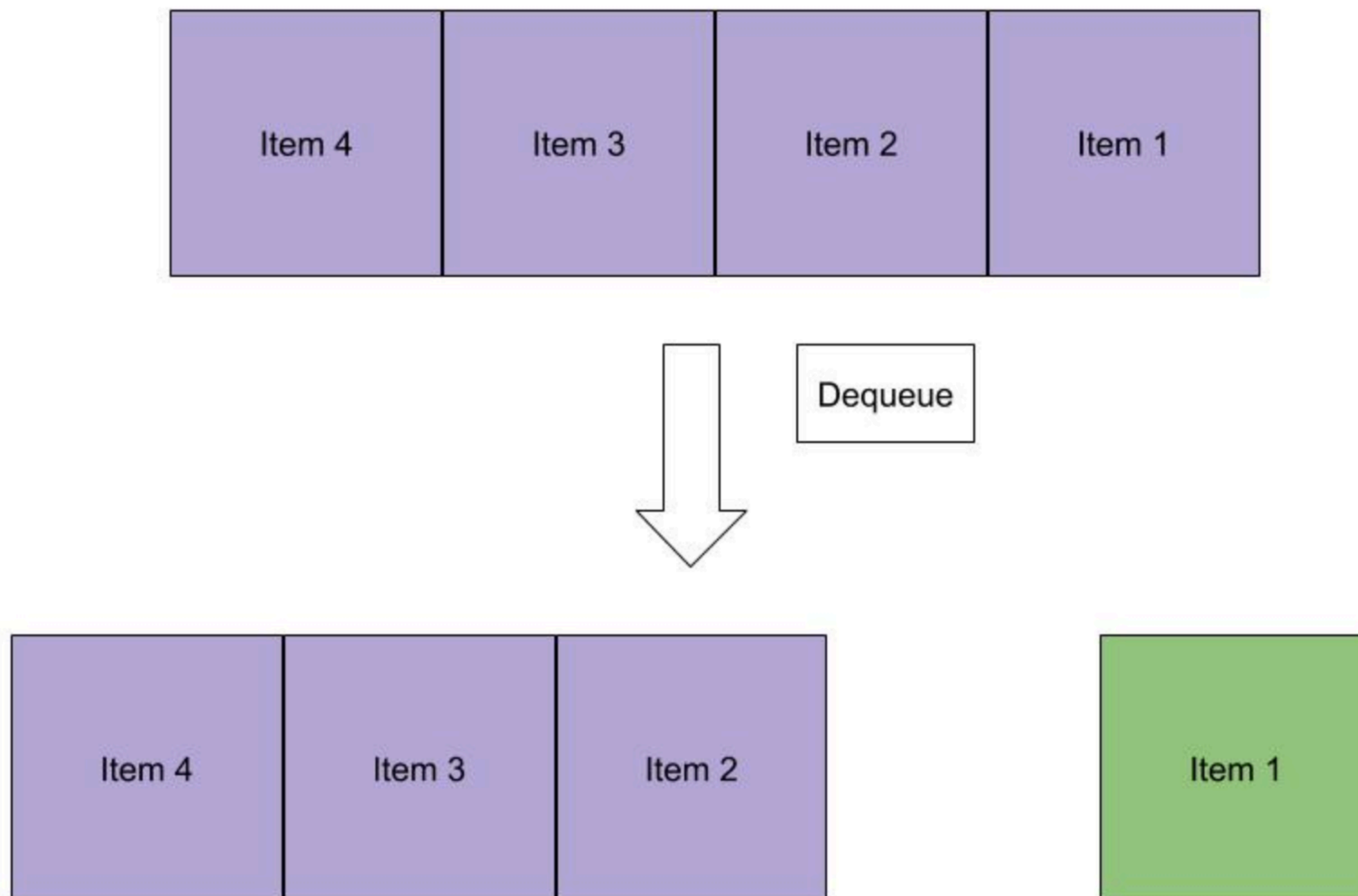
QUEUE

dátová štruktúra, ktorá sa riadi **First-in-First-Out** (FIFO) princípom



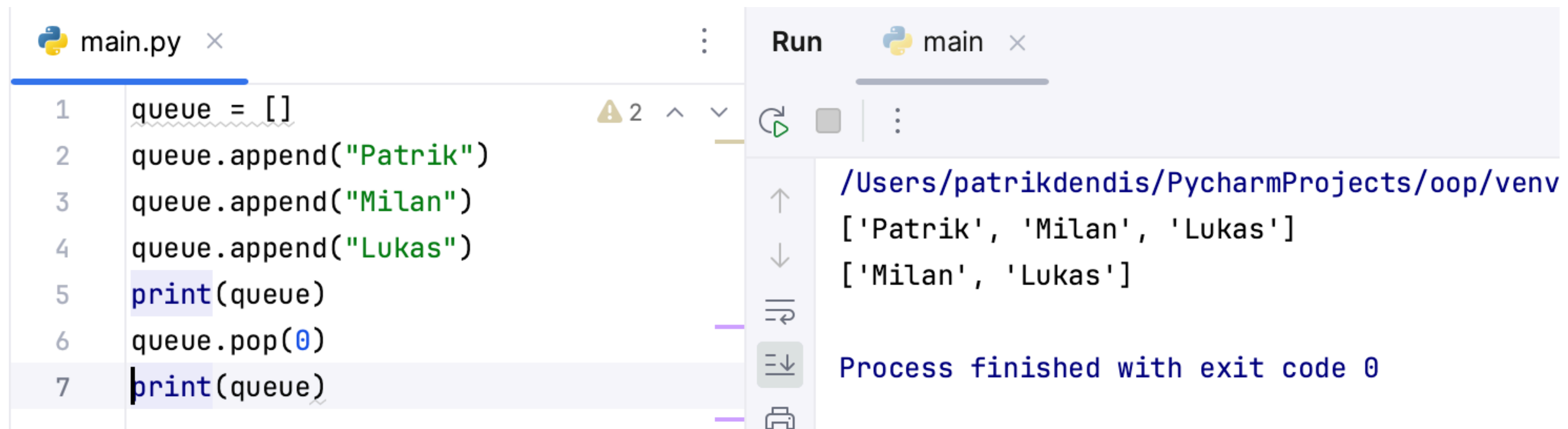
QUEUE

dátová štruktúra, ktorá sa riadi **First-in-First-Out** (FIFO) princípom



QUEUE

použiť list na simuláciu fronty pomocou metódy **append()** na pridanie prvku na koniec zoznamu a metódy **pop()** na odstránenie prvého prvku zo zoznamu



The screenshot displays a PyCharm IDE interface. On the left, a code editor window titled 'main.py' contains the following Python code:

```
1 queue = []
2 queue.append("Patrik")
3 queue.append("Milan")
4 queue.append("Lukas")
5 print(queue)
6 queue.pop(0)
7 print(queue)
```

On the right, a 'Run' window titled 'main' shows the execution output. The output consists of three lines:

```
/Users/patrikdendis/PycharmProjects/oop/venv
['Patrik', 'Milan', 'Lukas']
['Milan', 'Lukas']
```

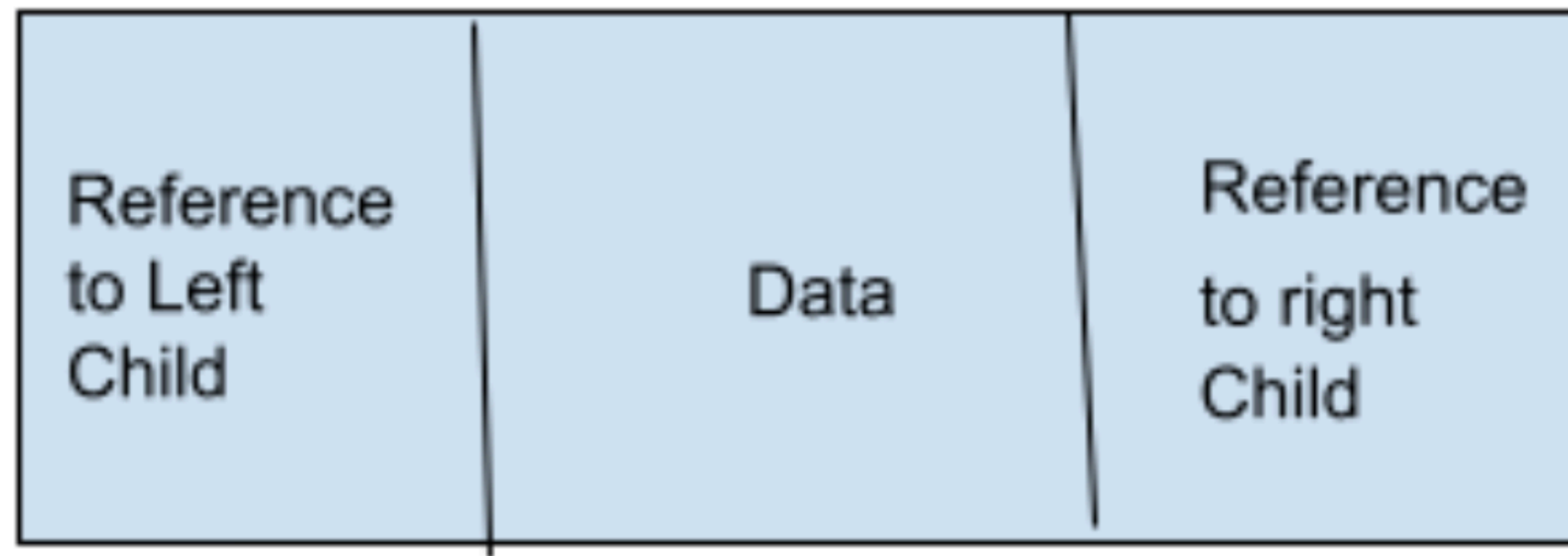
Below the output, a status message indicates: 'Process finished with exit code 0'.

POUŽITIE QUEUE

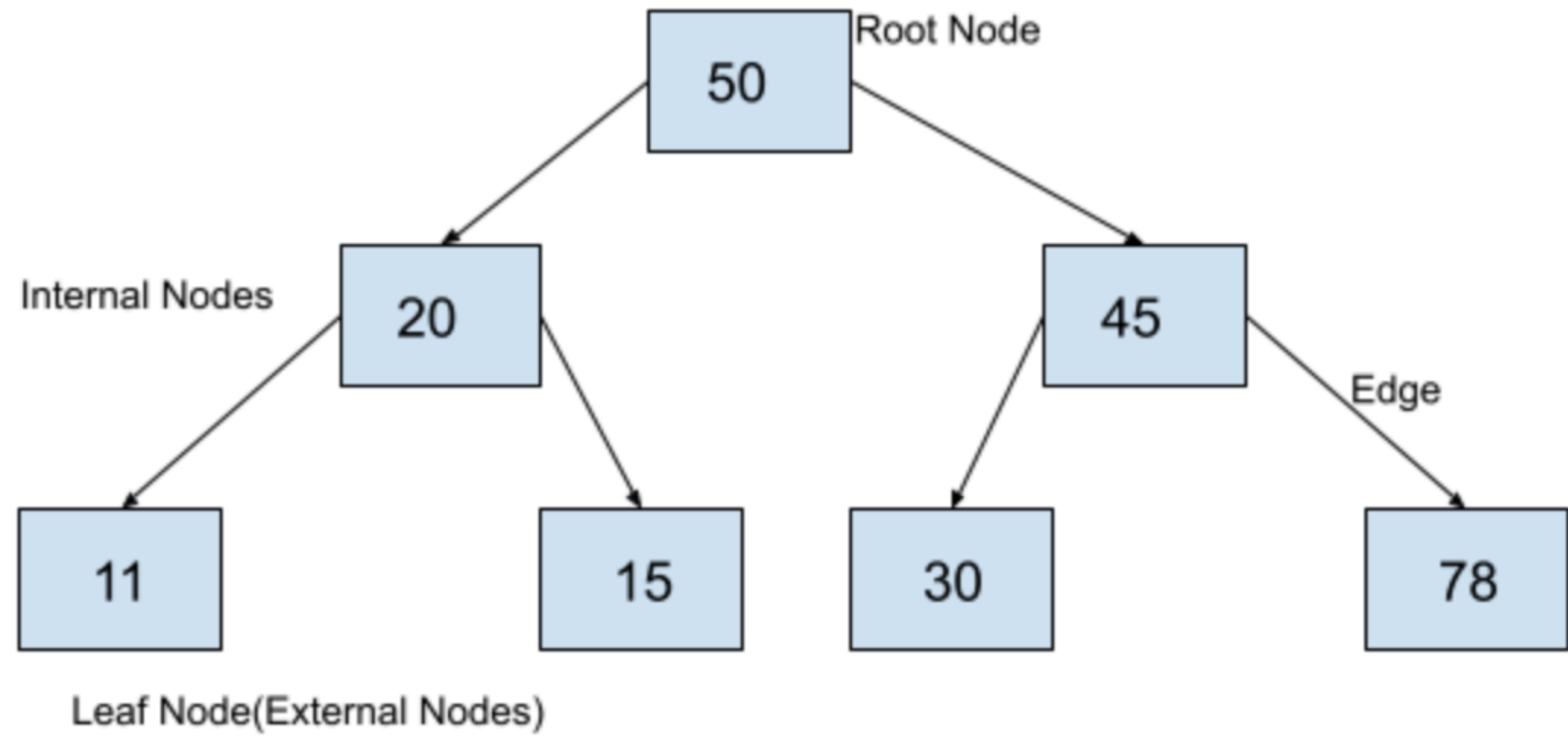
- tlačiareň používa frontu na ukladanie tlačových úloh. Prvá odoslaná úloha sa tlačí ako prvá
- systémy, ktoré vyžadujú spracovávanie udalostí v reálnom čase, ako sú hry alebo simulácie, môžu používať fronty na správu udalostí, ako sú vstupy od hráča alebo interné udalosti hry
- vo webových serveroch sa fronty používajú na spracovanie prichádzajúcich požiadaviek od klientov, spracovávajúc ich v poradí, v akom prišli

BINARY TREE

binárne stromy sú abstraktné dátové štruktúry prispôsobené na vyhľadávanie. Každý uzol stromu môže mať najviac dvoch potomkov, na ktorých ukazujú jeho ukazovatele. Okrem toho ešte uchováva hodnotu prvku.



BINARY TREE



BINARY TREE

	Binary Search Tree	
	Average	Worst
Insertion	$O(\log n)$	$O(n)$
Deletion	$O(\log n)$	$O(n)$
Searching	$O(\log n)$	$O(n)$
Traversal	$O(n)$	$O(n)$

POUŽITIE BINARY TREE

- sú ideálne pre aplikácie, kde sú časté vyhľadávacie operácie.
- binárne stromy sú ideálne pre situácie, kde dáta majú prirodzene hierarchickú štruktúru, ako sú súborové systémy alebo organizačné štruktúry.
- mnohé databázové systémy používajú varianty binárnych stromov, ako sú B-stromy a B+-stromy, pre efektívne indexovanie a rýchle vyhľadávanie.

ZADANIE

Create a queue class for character values. Implement the following operations:

- IsEmpty — check if the queue is empty;
- IsFull — check if the queue is full;
- Enqueue — add an element to the queue;
- Dequeue — delete an element from the queue;
- Show — display all queue elements on the screen.

When the app starts, display a menu that a user can use to choose the desired operation.

ZADANIE

Develop an app that stores information about user's credentials (username and password). Each user has a username-password pair. When the app starts, the following menu displays:

- Add a new user;
- Delete the existing user;
- Check if the user exists;
- Edit username of the existing user;
- Change password of the existing user.

Use one of the data structures for this task. Be guided by the task objective as you choose a data structure.

ZADANIE

The user inputs a set of numbers from the keyboard. The received numbers should be saved in a singly linked list. After this, display a menu offering the user the following items:

1. Add an item to the list.
2. Delete an item from the list.
3. Show the list contents.
4. Check if the list contains this value.
5. Replace a value in the list.

The action is performed based on the user's choice, and the menu displays again.

ĎAKUJEM ZA POZORNOST