Algoritmi in podatkovne strukture 1

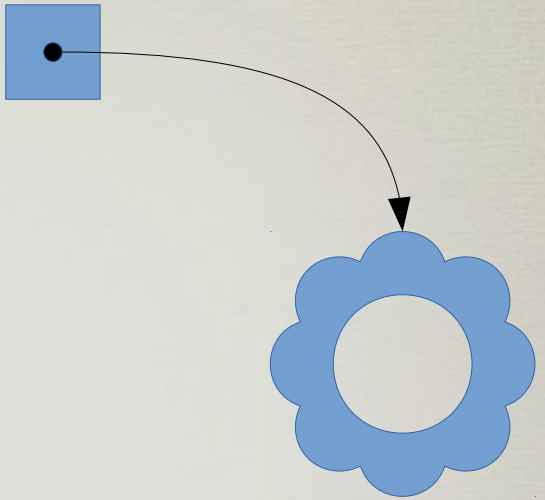
Visokošolski strokovni študij Računalništvo in informatika

Povezani seznami

Jurij Mihelič, UniLj, FRI

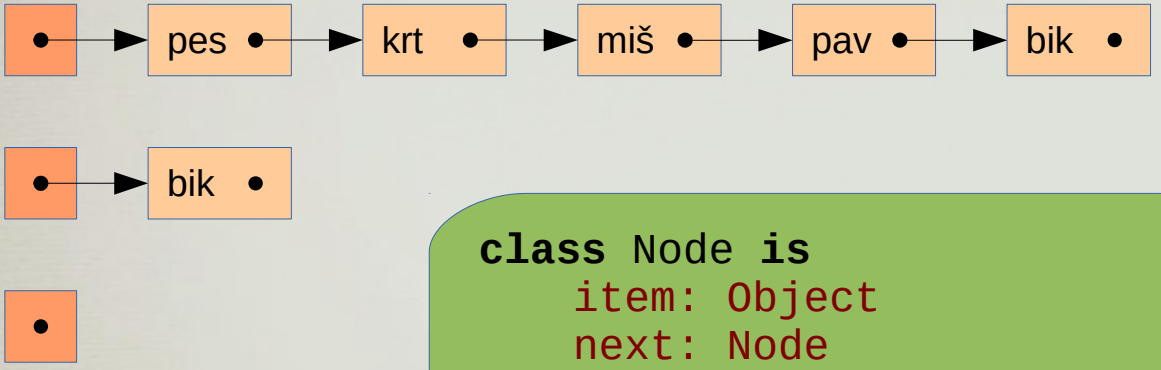
Povezani seznam

- Kazalčna podatkovna struktura
  - temelji na kazalcih
  - kazalec pove, kje se nekaj nahaja
  - vrednost kazalca je naslov
- Dinamična podatkovna struktura
  - njena velikost se lahko spreminja
- Vozlišče
  - vsebuje **element** oz. podatek in **kazalec**
  - **vozlišča** so med seboj povezana preko **kazalcev**



Enojno povezani seznam

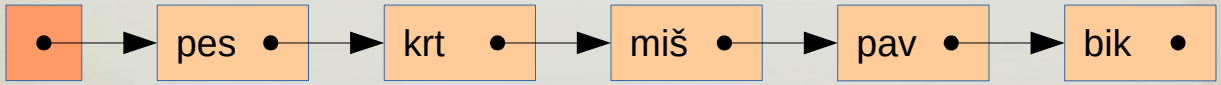
- Vozlišča vsebujejo povezavo v eno smer
  - na naslednji element
  - kateri je prvi in kateri je zadnji?




```
class Node is
    item: Object
    next: Node

    init Node(item, next) is
        this.item = item
        this.next = next
    end
end
```

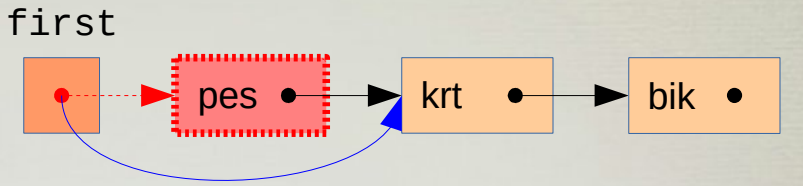
Enojno povezani seznam

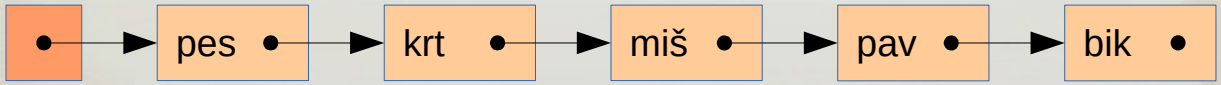
- Ustvarjanje seznama
  - ustvarjanje vozlišč
- Sprehodi po seznamu
  - prehod seznama
  - iskanje vozlišča po vrednosti
  - iskanje zadnjega vozlišča



Enojno povezani seznam

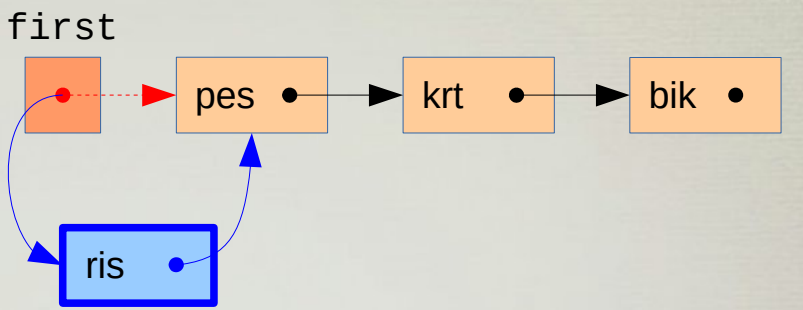
- Odstranjevanje elementov
  - vseh
  - prvega
  - drugega
  - za podanim

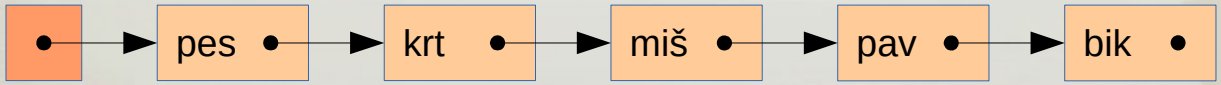




Enojno povezani seznam

- Dodajanje elementov
  - pred prvim
  - za prvim
  - za podanim
  - za zadnjim
  - pred podanim

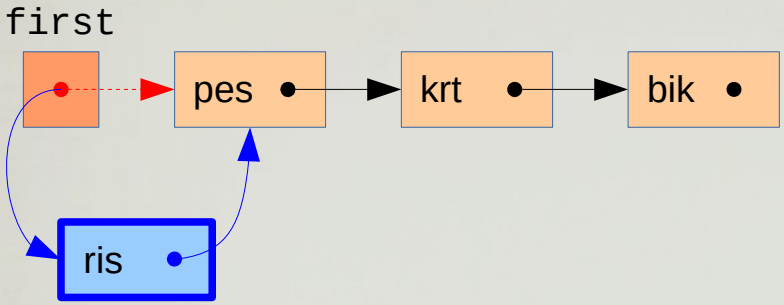




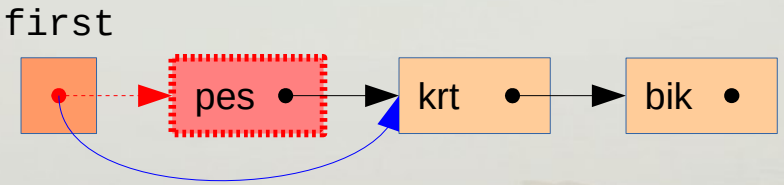
Enojno povezani seznam

- EPS kot sklad

```
push("ris")
```

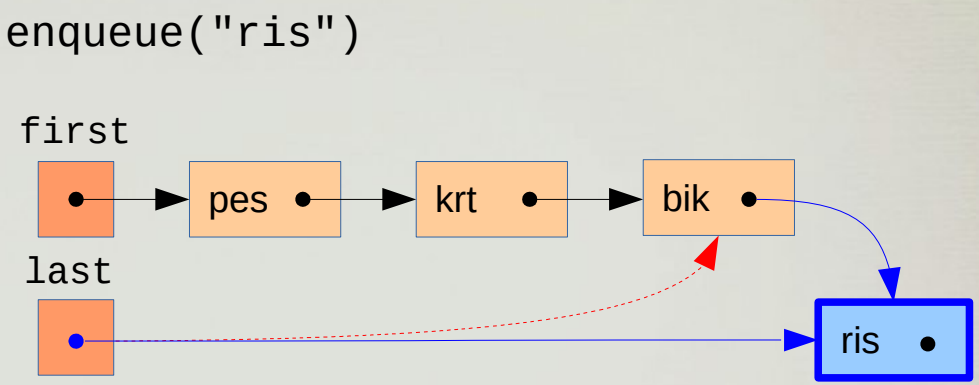


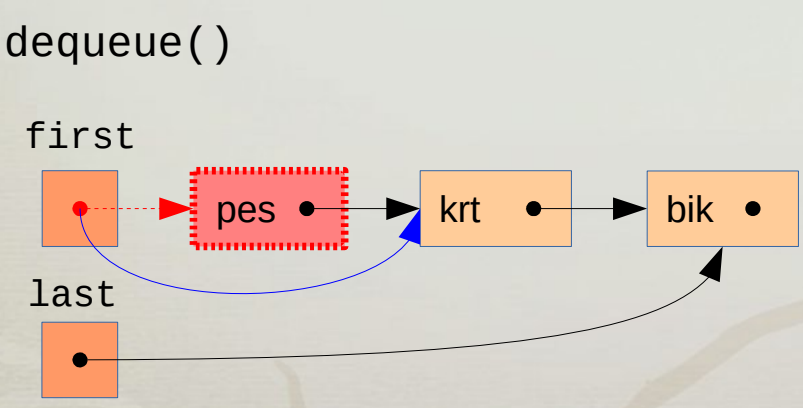

```
pop("ris")
```



Enojno povezani seznam

- EPS kot vrsta
  - dodamo atribut `last`





Dvojno povezani seznam

- Vsako vozlišče ima dve povezavi:
  - naprej oz. naslednji
  - nazaj oz. prejšnji

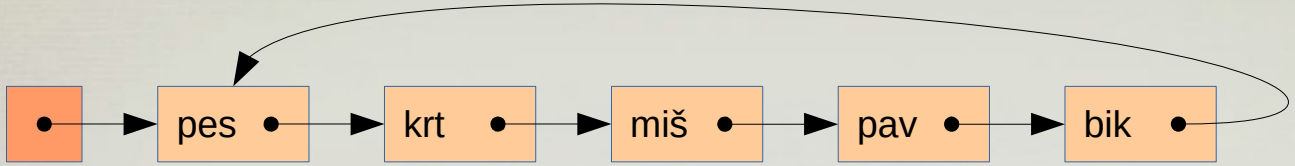

```
class Node is
    item: Object
    prev: Node
    next: Node

    init Node(item, prev, next) is
        this.item = item
        this.prev = prev
        this.next = next
    end
end
```

Dvojno povezani seznam

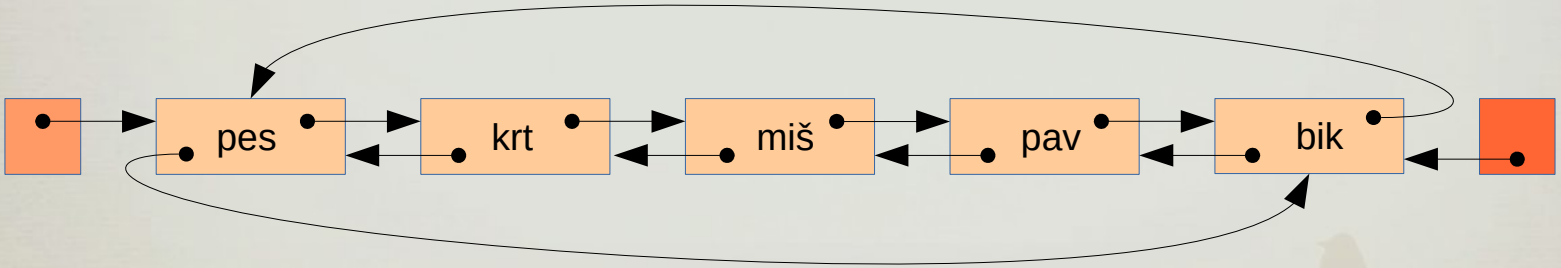
- Operacije
  - brisanje spredaj in zadaj
  - dodajanje spredaj in zadaj

Ciklično povezani seznam

- Ciklični EPS

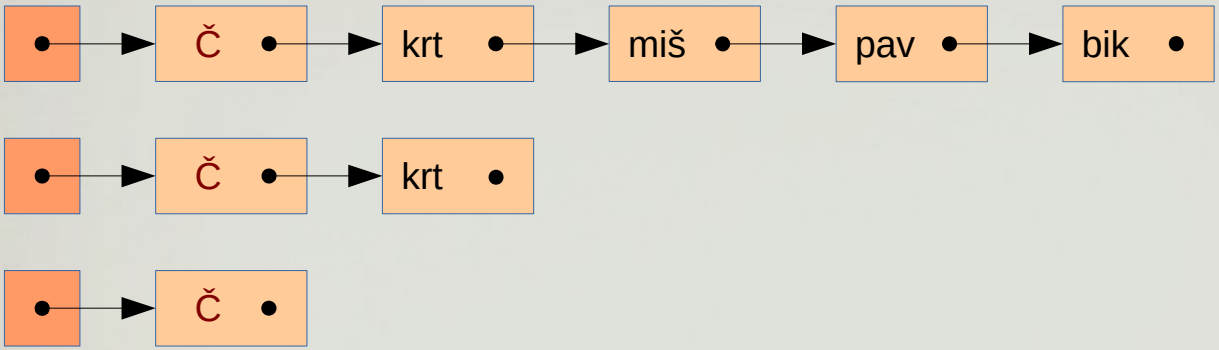


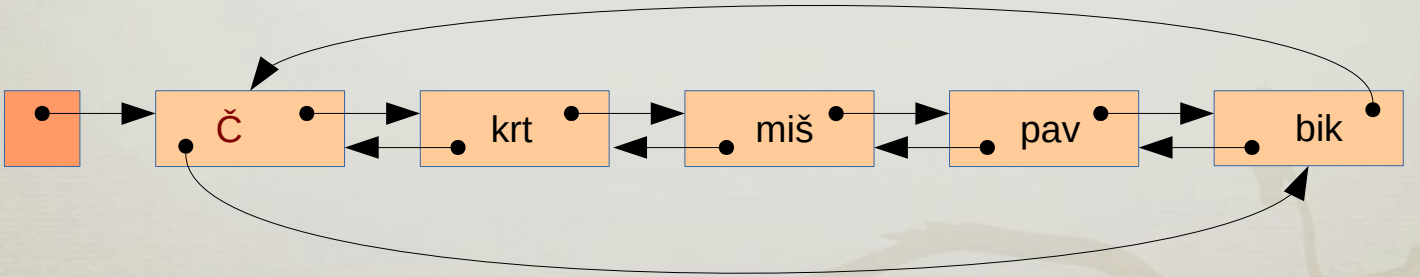
- Ciklični DPS



Povezani seznam s čuvajem

- Čuvaj (*sentinel*) *praznega seznama*
  - poseben element, ki je vedno prisoten





Seznam v polju

- Enojno povezani seznam
  - polje `next` vsebuje indekse naslednjih elementov
  - polje `item` vsebuje elemente
  - atribut `first`
- Prosta mesta
  - atribut `free`
  - `allocate()`
  - `free(i)`
- Dvojno povezani seznam
  - dodatno polje `prev`

`first = 5`

0	1	2	3	4	5	6	7	8	9
4	3	0	7	9	1	8	6	-1	-1
	b		c		a	e	d	f	

`free = 2`

Persistenca

- Hranjenje zgodovine sprememb
  - enojno povezani seznam
  - operaciji `push(x, first)` in `pop(first)` vrneta nov seznam (oz. kazalec na prvi element seznama)

```
fun push(l, x) is
    return Node(x, l)

fun pop(l) is
    return l.next
```

Implicitne in eksplicitne PS

- Implicitna podatkovna struktura
  - porabi *zelo malo* dodatnega prostora poleg prostora za shranjene podatke
  - odnosi med elementi so shranjeni implicitno
    - z vrstnim redom elementov v pomnilniku
    - npr. zaporedje je podano z vrstnim redom v polju
- Eksplicitna podatkovna struktura
  - potrebuje dodatni prostor za hranjenje odnosov
  - odnosi med elementi so shranjeni eksplicitno
    - pogosta uporaba kazalcev
    - npr. zaporedje je podano z relacijo *naslednik* (kazalec)

Povzetek

	operacija	EPS	DPS
sklad, vrsta, dvrsta	enqueue(x)	$O(1)$	$O(1)$
	dequeue(), pop()	$O(1)$	$O(1)$
	enqueueFront(x), push(x)	$O(1)$	$O(1)$
	dequeueBack()	$O(n)$	$O(1)$
zaporedje	get(i)	$O(n)$	$O(n)$
	set(i, x)	$O(n)$	$O(n)$
	find(x)	$O(n)$	$O(n)$
	insert(i, x)	$O(n)$	$O(n)$
	delete(i)	$O(n)$	$O(n)$
vreča, množica	remove(x)	$O(n)$	$O(n)$
	add(x) – vreča	$O(1)$	$O(1)$
	addUnique(x) – množica	$O(n)$	$O(n)$