

# DELTA TopGun

## (05) Úvod do abstraktních datových struktur - Spojový seznam

Tomáš Faltejsek, Luboš Zápotočný, Michal Havelka

2022

# Abstraktní datové struktury

- Rozšiřují možnosti programového řešení
- Rozšiřují aplikovatelnost informatiky na řešení problémů
- Reprezentace "přirozeně" (v *kompilátorech*) neexistujících datových struktur
- Jsou definována specifickým setem pravidel – od tud *abstraktní*
  - → **co** má datová struktura dělat, ne **jak** to má dělat
- Pro člověka srozumitelnější reprezentace

# Abstraktní datové struktury

- Seznamy (*Lists*)
- Binární stromy (*Binary trees*)
- Grafy (*Graphs*)
- ...

# Abstraktní datové struktury

- **Seznamy (Lists)**
- Binární stromy (*Binary trees*)
- Grafy (*Graphs*)
- ...

# Spojový seznam - jednosměrně zřetězený

## Node

- Nese hodnotu *data* a ukazatel *next*
- **data**
  - Obsahují hodnotu nebo odkaz na hodnotu (int, char, pointer, ...)
- **next**
  - Ukazatel na následující prvek (*Node*) v seznamu
  - Ukazatel *next* v případě konce seznamu odkazuje na *NULL*

## INFO

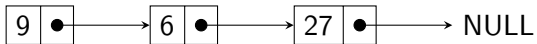
- Záznam informačního charakteru - drží dva ukazatele
- **head** - odkaz na *Head* (první *Node* seznamu)
- **tail** - odkaz na *Tail* (poslední *Node* seznamu)

# Struktura Node v jazyce C

```
typedef struct Node {  
    int data;  
    Node* next;  
} Node;
```



# Jednosměrně zřetězený



# Průchod spojovým seznamem (*traversal*)

```
void printList(Node * node) {  
    while (node->next != NULL) {  
        printf("%d□\n", node->data);  
        node = node->next;  
    }  
}
```



# Spojový seznam vs. pole

- Pole: navazující blok alokované paměti ( $\text{sizeof}(\text{datatype}) * \text{size}$ )
- LinkedList: elementy mohou být uloženy kdekoliv v paměti
- Složitost x insert/delete operací

## Vložení prvku na první pozici (*prepend*)

- 1 Vytvoření nového uzlu (*node*)
- 2 Nastavení ukazatele *next* nového uzlu na aktuální *head* uzel
- 3 Nastavení ukazatele *head* na nově vložený uzel

## Vložení prvku na první pozici (*prepend*)

- 1 Vytvoření nového uzlu (*node*)
- 2 Nastavení ukazatele *next* nového uzlu na aktuální *head* uzel
- 3 Nastavení ukazatele *head* na nově vložený uzel

### Speciální případ

Může nastat situace, kdy vkládaný prvek bude jediným prvkem v seznamu. V takovém případě je nutné nastavit:

```
newNode->next = NULL;  
INFO->head = newNode;
```

# Spojový seznam vs. pole – vložení prvku

## Otázka

Jak vložit element na n-tou pozici v poli?

# Spojový seznam vs. pole – vložení prvku

## Otázka

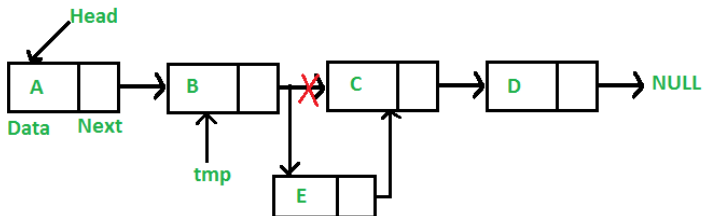
Jak vložit element na  $n$ -tou pozici v poli?

- U spojového seznamu musíme provést traversal k  $n - 1$  prvku a pak uzly "přepojit"
- U pole musíme všechny elementy po místě vložení posunout (*shift*) o  $+1$  index

## Speciální případ

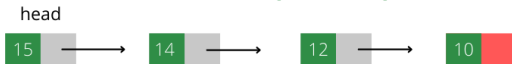
Při vložení před head nebo za tail, je nutné "přepojit" i head/tail ukazatel

# Spojový seznam vs. pole – vložení prvku

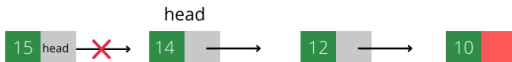


# Spojový seznam - smazání prvku ve spojovém listu

`deleteNode(head, 14)`



`deleteNode(head->link, 14)`



`head = head -> link`



# Smazání celého spojového seznamu najednou

– **Demonstrace na tabuli** –



# Přehled náročnosti operací

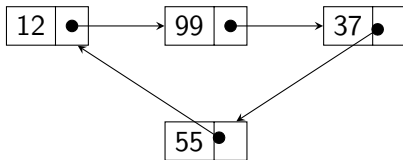
– **Demonstrace na tabuli** –

## Spojový seznam – oboustraně zřetězený

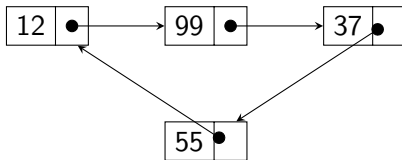
- Navigace dopředu/zpět v prohlížeči
- Undo/redo ve Wordu

– **Demonstrace na tabuli –**

# Spojový seznam - kruhově zřetězený

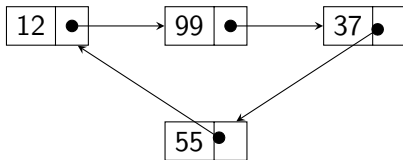


# Spojový seznam - kruhově zřetězený



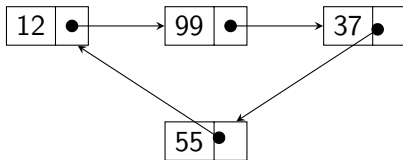
- Poslední blok vždy odkazuje na první

# Spojový seznam - kruhově zřetězený



- Poslední blok vždy okazuje na první
- Lze rozšířit na obousměrně kruhově zřetězený seznam

## Spojový seznam - kruhově zřetěžený



- Poslední blok vždy okazuje na první
- Lze rozšířit na obousměrně kruhově zřetěžený seznam
- Operace jako vkládání, mazání jsou stejné

# Spojový seznam - aplikace

# Spojový seznam - aplikace

- Procházení v historii webového prohlížeče



# Spojový seznam - aplikace

- Procházení v historii webového prohlížeče
- Implementace fronty

# Spojový seznam - aplikace

- Procházení v historii webového prohlížeče
- Implementace fronty
- round-robin scheduling

## Spojový seznam - aplikace

- Procházení v historii webového prohlížeče
- Implementace fronty
- round-robin scheduling
- Reprezentace grafů

# Spojový seznam - aplikace

- Procházení v historii webového prohlížeče
- Implementace fronty
- round-robin scheduling
- Repräsentace grafů
- Konstrukce Fibonacciho haldy

# Spojový seznam v kernelu Linuxu

`mm_struct`

`vm_area_struct`

# Spojový seznam v kernelu Linuxu

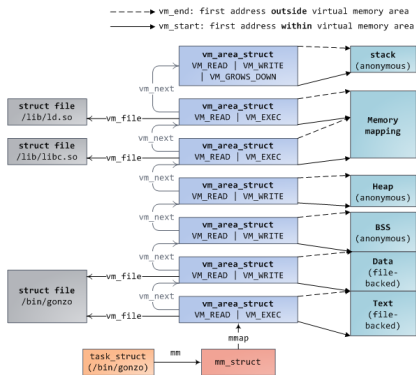


Figure: Virtual memory layout

<https://manybutfinite.com/post/how-the-kernel-manages-your-memory>