

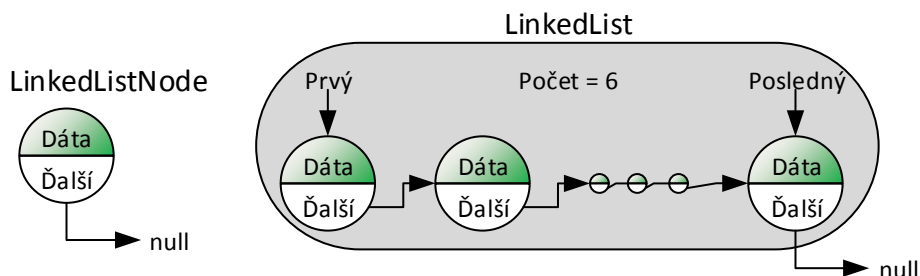
## LinkedList

LinkedList je implementáciu zoznamu pomocou navzájom zreťazených prvkov (v našom prípade LinkedListNode). Každý prvok má odkaz na ďalší prvok zoznamu a dáta. Zoznam je teda tvorený reťazou objektov typu LinkedListNode. Atribúty LinkedListu potom sú:

- Odkaz na **prvý** prvok (typu LinkedListNode) v zreťazení,
- Odkaz na **posledný** prvok (typu LinkedListNode) v zreťazení a
- **Počet prvkov** v zozname.

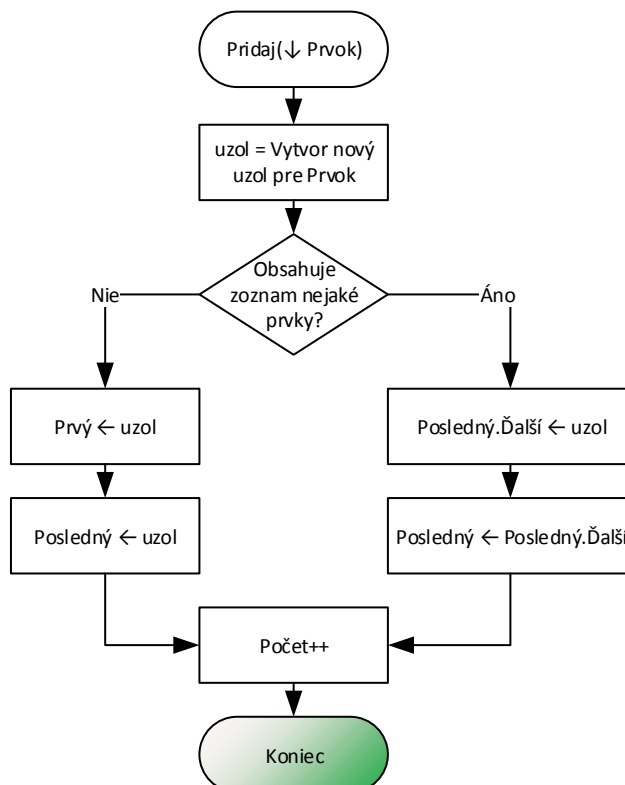
Po vytvorení LinkedList neobsahuje žiadny prvok, teda odkazy na prvý a posledný prvok ukazujú na null a počet prvkov je rovný 0.

Každý objekt typu LinkedListNode dokáže sprístupniť a modifikovať svoje položky Dáta a Ďalší. Konštruktor tejto triedy ako parameter preberá Dáta (nemá zmysel vytvárať LinkedListNode, ak nie sú k dispozícii dáta), ktorými sa inicializuje. Odkaz na ďalší prvok je inicializovaný na null.

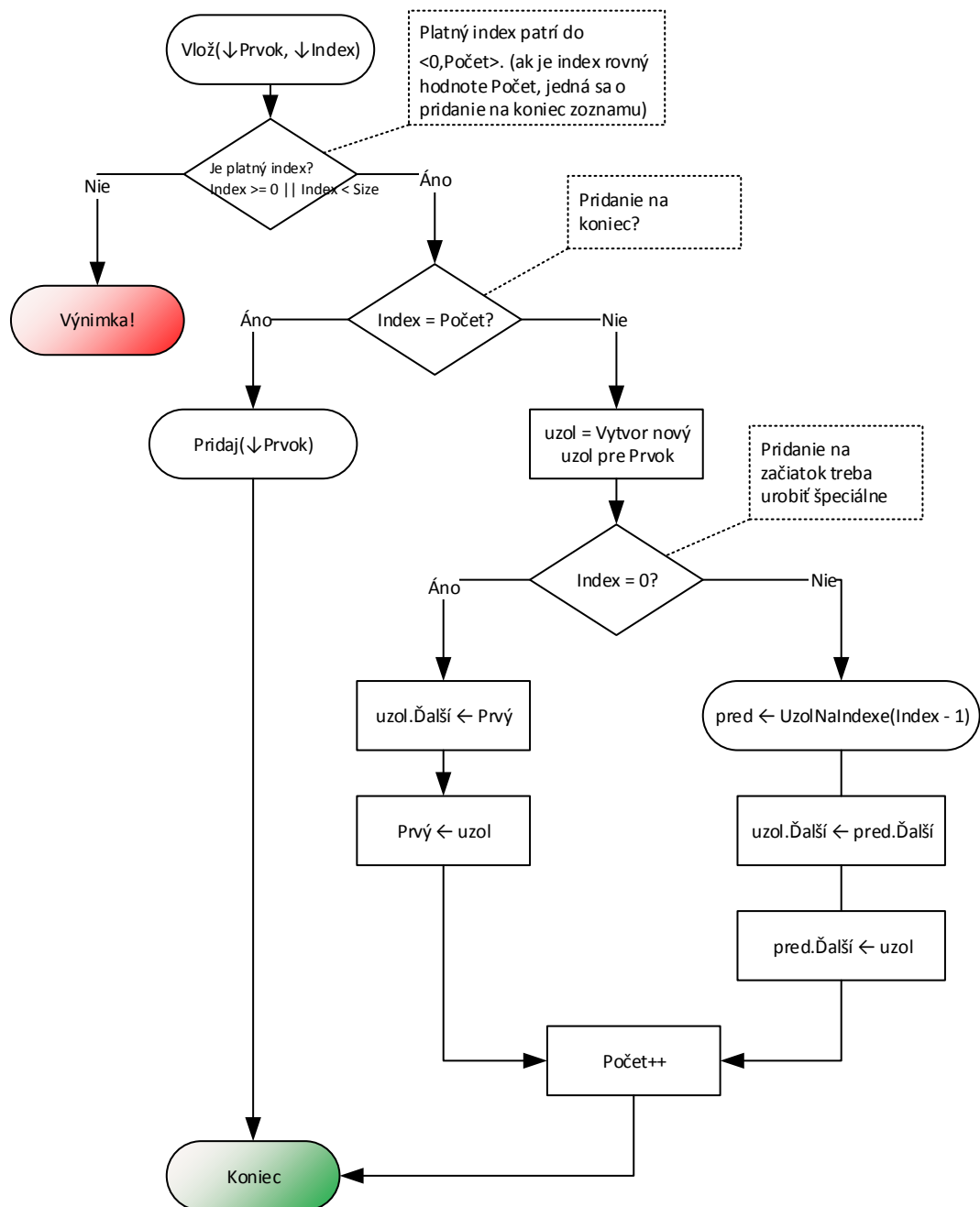


## Metódy

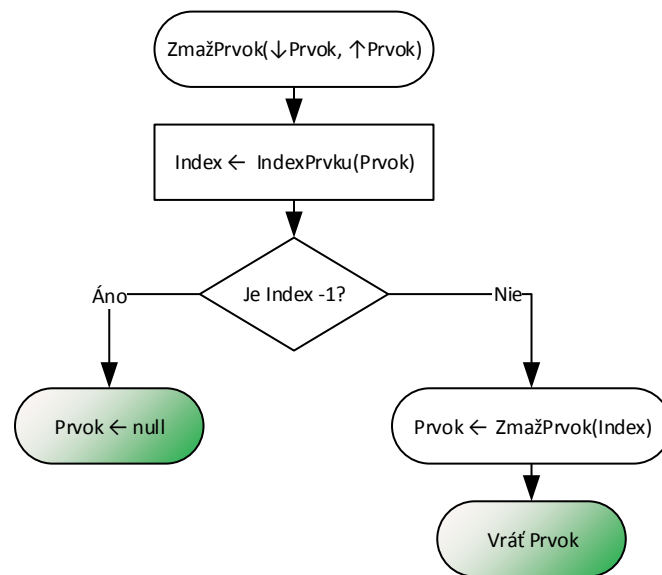
**Pridaj prvok** pridá prvok na koniec zoznamu.



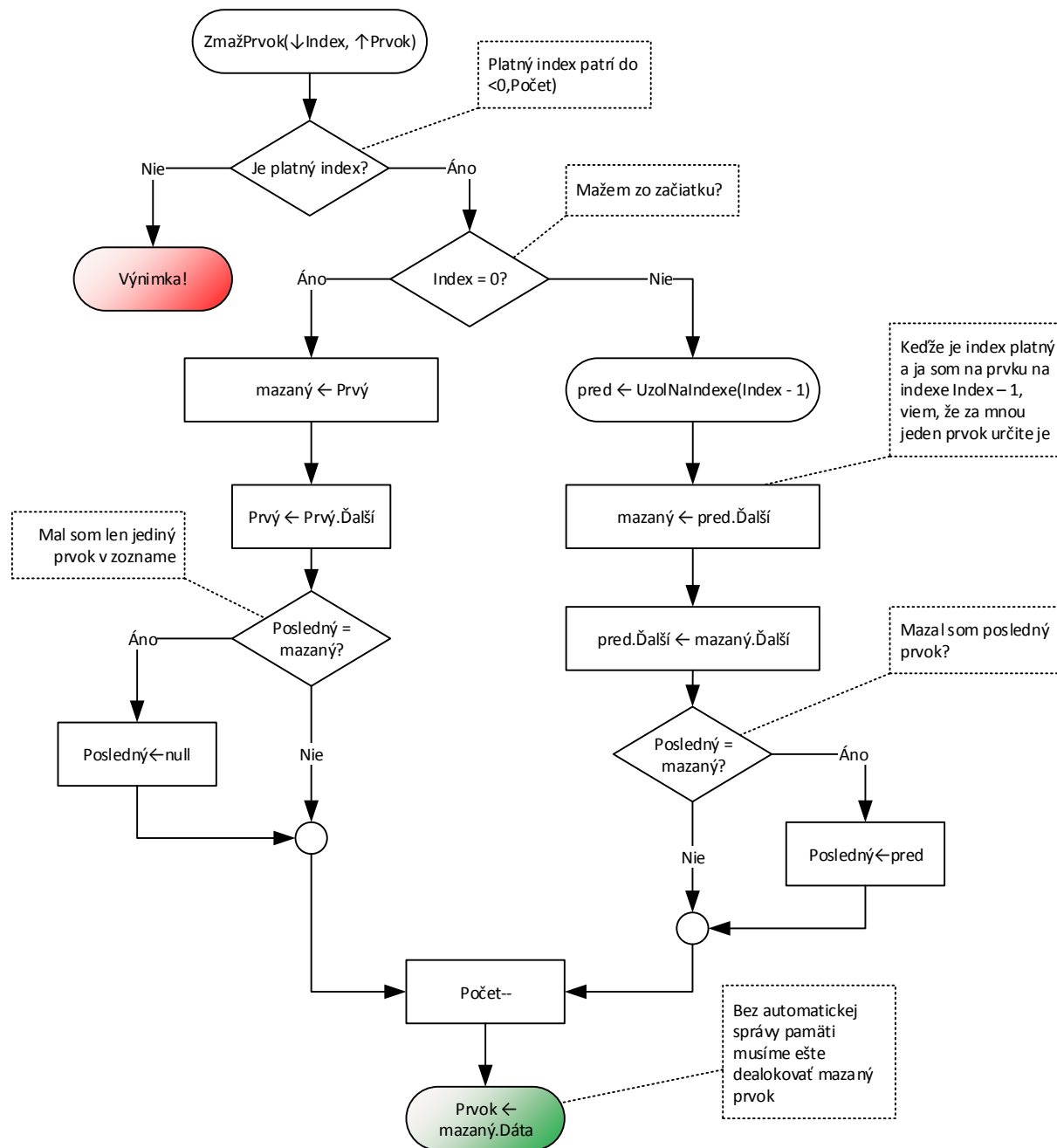
**Vlož prvok na daný index** vloží (neprepiše) prvok na daný index. Ak je index zhodný s počtom prvkov (teda jedná sa o prvý neplatný index v poli), metóda sa správa ako obyčajné vloženie prvku.



**Zmaž prvok** nájde prvok v zozname a ten následne vymaže (vymaže sa iba prvý výskyt daného prvku). Na znak úspešného vymazania vráti vymazaný prvok. Ak sa prvok v zozname nenachádza metóda vráti null.

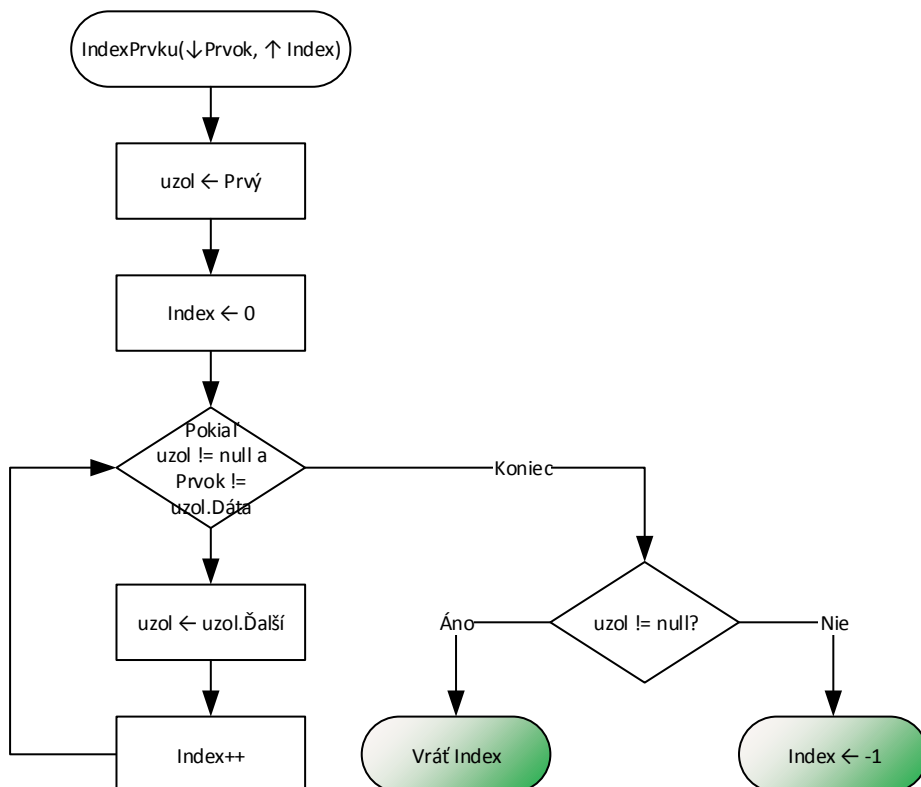


**Zmaže prvok na danom indexe** (ak nie je index platný, vyhodíme výnimku). Odstránený prvok tvorí návratovú hodnotu tejto metódy (nikdy<sup>1</sup> nad ním nevoláme deštruktor). Naopak, ak nemáme automatickú správu pamäti, nesmieme zabudnúť zavolať deštruktor odstráneného uzla.

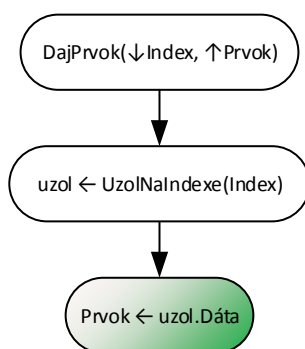


<sup>1</sup> Pokiaľ to nie je požadovaná vlastnosť a očakávané správanie štruktúry.

**Vráti index prvku**, ak sa v zozname nachádza. Vždy vráti index prvej hodnoty (ak sa v zozname nachádza viac rovnakých hodnôt). Referenčné typy porovnávame referenčne<sup>2</sup>! Ak sa prvok v zozname nenachádza, vráti -1.



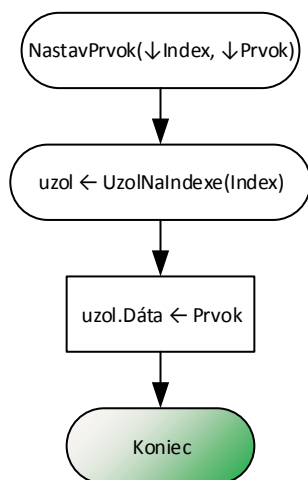
**Vráti prvok na danom indexe**. O kontrolu správnosti indexu sa stará metóda `UzolNaIndexe`, ktorá v prípade neplatného indexu vystrelí výnimku.



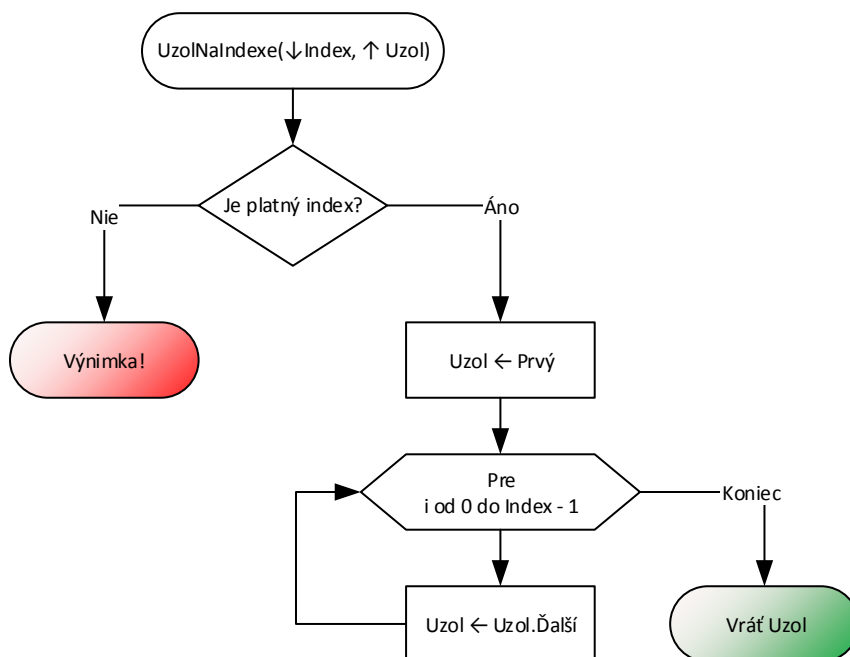

---

<sup>2</sup> ==

**Prepíše prvok na danom indexe.** O kontrolu správnosti indexu sa stará metóda UzolNaIndexe, ktorá v prípade neplatného indexu vystrelí výnimku.

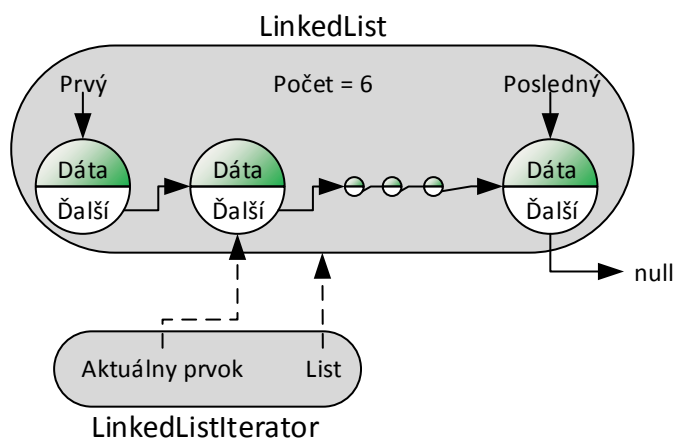


Pomocná metóda, ktorá **vráti uzol na zadanom indexe**. Ak je index neplatný, metóda vystrelí výnimku.

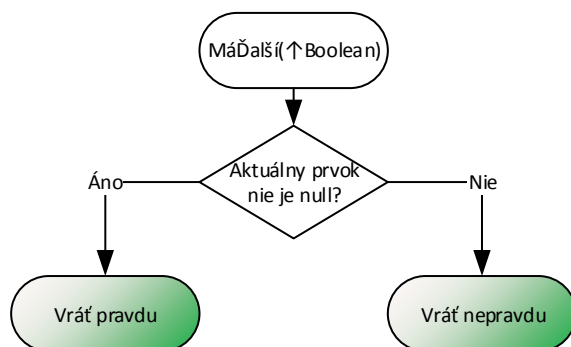


## Iterátor

Iterátor pre LinkedList sa v konštruktore naviaže na zoznam, cez ktorý má iterovať. Obsahuje atribút Aktuálny prvok typu LinkedListNode, ktorý sa inicializuje na prvý prvok v zozname<sup>3</sup>.



**Vráti príznak, či má LinkedList ešte ďalší prvok** (teda Aktuálny prvok ukazuje na ďalší prvok).



<sup>3</sup> Tým, že je inicializovaný na prvý prvok, musíme chápať metódu **next** tak, že najskôr sa vráti hodnota prvku, na ktorý sa ukazuje a až potom sa iterátor posunie – inak by sa prvý prvok preskočil. Nemôžeme tu postupovať tak, ako v **ArrayListe**, že sa najskôr posunieme a potom vrátime hodnotu (čo je správne správanie), pretože sa nevieme odkázať „pred“ zoznam, ako sme to urobili v **ArrayListe**.

**Vráti prvok a posunie sa v štruktúre.** Platnosť netreba kontrolovať, keďže tá bola skontrolovaná metódou MáĎalší. O jej korektné volanie sa stará JAVA sama.

