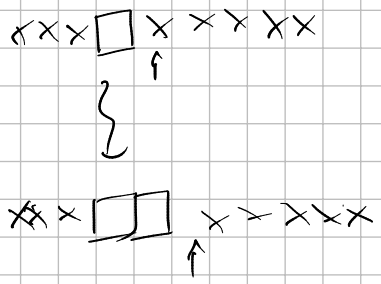
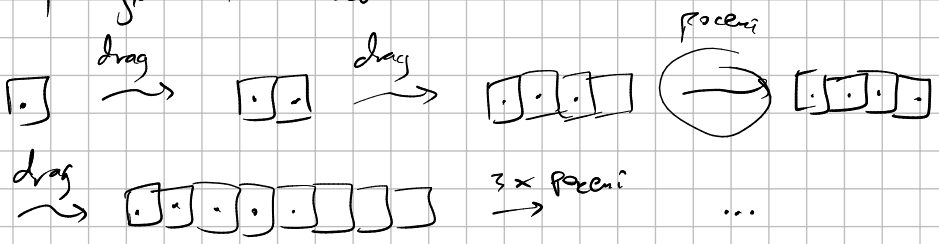
Podatkovne strukture

- vector → dodajanje na konec je $O(1)$.



naivno dodajanje $O(n)$

Ideja: Namesto da vedno dodam en prostor, podvojim trenutno velikost



↳ push_back() → dodaj na konec vektorja
$O(1)$ amortizirano
v povprečju

↳ size()

↳ pop_back()

↳ [ ]

↳ ...

SKLAD: poseben seznam, kjer lahko delamo samo naslednje operacije:

- → dodaj na vrh
- → odvzemo iz vrha
- → pogled vrh
- → velikost

3

Opazujmo naslednje operacije:

+3, +7, +9, +1, −, −, +4, −, +5, −, −

Temu rečemo LIFO – Last in, first out

To je samo vektor, v katerem ni dovoljena uporaba [] (ne smemo dostopati do poljubnega elementa)

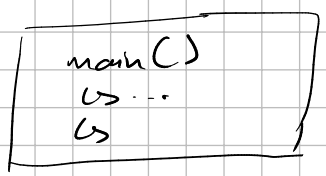
```
int f(int x){
    int y = 7*x;
    y += g(y-1);
    return y;
}
```

```
int g(int x){
    return x+1;
}
```

f izvaja:
int x;
int y;

→

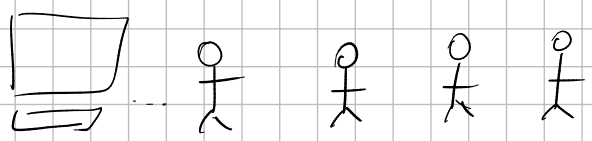
g izvaja:
int x



VRSTA : FIFO - first in, first out

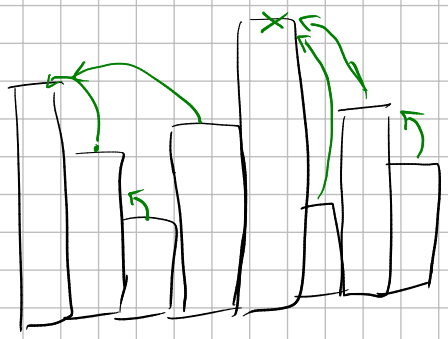
Neka sorta seznama, kjer imamo dovoljene le naslednje operacije:

- → dodamo na konec
- → odvzamemo iz začetka
- → pogledamo na začetek



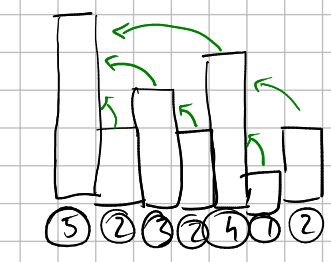
Glavna ideja je, da „prvi pride, prvi melje" - podatki „čakajo na nekaj", in prvi, ki smo ga videli, bo prvo obravnavan

+3, +7, +9, +1, −, −, +4, −, +5, −, −



Želeli bi, da za vsako stolpnico najdemo najbližjo stolpnico levo od nje, ki je nižja od nje

Ideja rešitve: Hranimo si padajoč sklad višin stolpnic, ki smo jih videli do zdaj. Ko najdemo stolpnico, če je nižja od najnižje na skladu, potem jo dodamo na sklad, sicer pa iz sklada jemljemo, dokler se to ne zgodi



2
4
5

$O(n)$ rešitev, ker vsako stolpnico damo največ enkrat na sklad

MNOŽICE IN SLOVARJI

$\{1, 3, 10\}$

Želimo si hraniti (na dovolj učinkovit način) neko množico podatkov. Cilj je, da lahko hitro dodajamo, hitro odvzemamo in hitro preverimo, če je neki element v množici.

↳ insert()
↳ erase()
↳ count() ← vrne 0 ali 1

} $O(\log n)$ HITRO!

	SEZNAM	UREJEN SEZNAM	MNOŽICA	NEUREJENA MNOŽICA
dodaj	$O(1)$	$O(n)$	$O(\log n)$	$O(1)$
briši	$O(n)$	$O(n)$	$O(\log n)$	$O(1)$
poišči	$O(n)$	$O(\log n)$	$O(\log n)$	$O(1)$
i-ti element	$O(1)$	$O(1)$	X	X