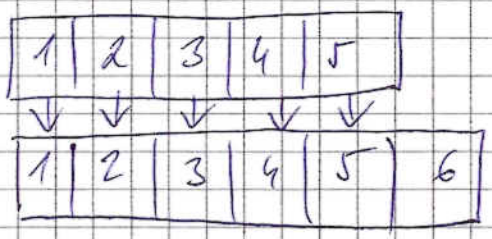
DATOVÁ STRUKTURA POLE NEOMEZENÉHO

2. 11. 2010

pole [ ]
pole.pushBack (e) = vlož prvek
e = pole.popBack() = vyjmi prvek

zvětšit pole = přidat prvek + původní zkopírovat



nebo dělná metoda,
při každém přidání kopíruj vše

řešení - přidat vždy pole než jen o jedno

ALE

je nemysl držet zase extrémně dlouhé pole
=> nutno zvažit poměr přidávaných polí

```
int pole[] = new pole[1];
int pocet = 0;
int delka = 1;
void pushBack(e){
    [illegible]
    if (pocet == delka){
        [illegible]
        pole = realocate(pole, 2*delka, pocet)
        delka = delka * 2
    } // if
    pole[pocet] = e
    pocet++
} // pushBack
```

```
int popBack() {
    pocet = pocet - 1;
    int x = pole[pocet];
    if (pocet <= delka/4) {
        pole reallocate(pole, delka/2, pocet)
        delka = delka/2
        } // if
    return x;
} // popBack

int [] reallocate(pole, delka, pocet) {
                  pole prvku   ints

    int *pole2[] = new int pole[delka];
    for (i = 0; i < pocet; i++) {
        pole2[i] = pole[i];
        } // for
    return pole2;
    } //reallocate
```

JAK DLOUHO TO TRVÁ? =>
JAK SPOČÍTAT SLOŽITOST?

přidání / odebrání prvku = trvá „n"
ale přidání / odebrání je v programu jen občas

=> použijeme tzv. AMORTIZOVANOU SLOŽITOST
= na dobré kroky dáváme + } tzv. tokeny
= na blbý kroky dáváme − }
a potřebuji se dostat ze nějaké ∅, NESMÍM do kapsy !

dle Melichorna

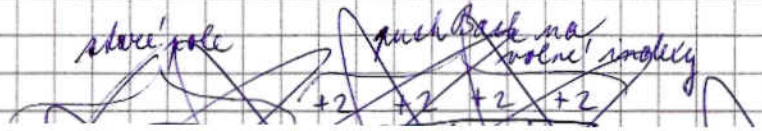
pushBack = + 2 tokeny
popBack = − 2 tokeny
~~realloc~~ realloc = − 1 tokem na každý kopírovaný prvek = na každý průchod cyklem

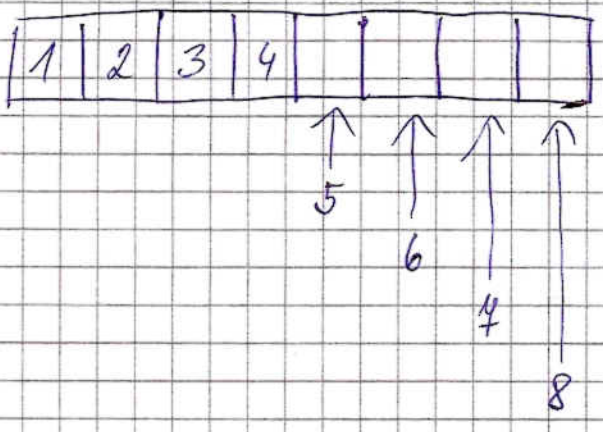
staré pole - délka d', počet = d'

2 * d' .... d' volné + 2 * d'

2 * (2 * d') .... 2d' prvků přesunu
↑ potřebuji (−1 * 2d' tokenů)

součet = 0 tokenů

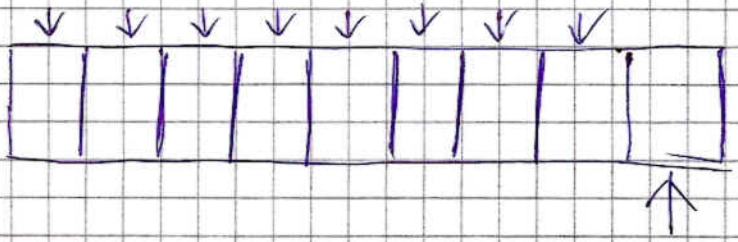




= 0 tokenů

+2
+2
+2
+1 = 8 tokenů

pro pushBack

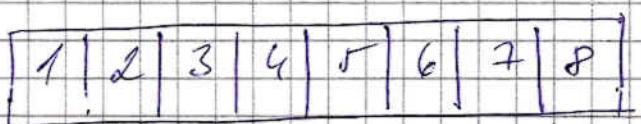


pro reallocate

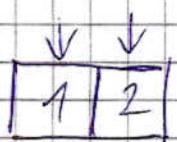
−8 = 0 tokenů

9 +2 = 2 tokeny = OK!

pro popBack



odebrání +1
+1
⋮
= 6 tokenů



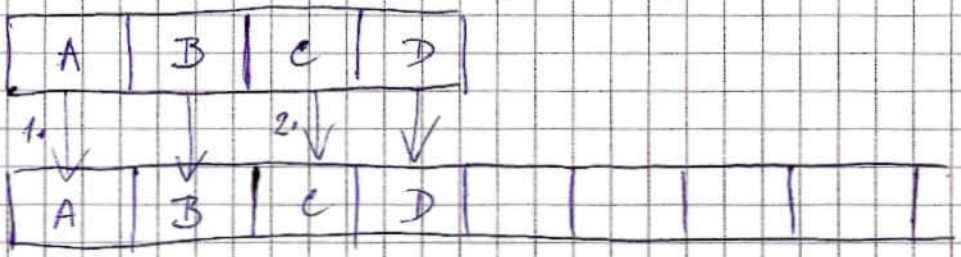
min. délka = délka/4

→ realloced −2 tokeny

zbytek = 4 tokeny
= OK!

Nutno vždy nasbírat ⊕ tokeny, abych bylo ~~se~~ z čeho vydělat.

realokace pro realloc =>
založím 2 pole a postupně kopíruji skládané prvky



kopíruje se vždy po složení 2 prvků
takto je pro konstantní

(Hw) Cv 3.19

Naprogramovat frontu pomocí dvou zásobníků
= FIFO.

---

překlad kpt 4 = „Hardware"