Radix sort

Jos jedan algoritam koji ne koristi upoređivanje kao 'osnovni alat' je radix sort.

Najpre uvedimo pojam stabilnog algoritma za sortiranje. Kažemo da je algoritam za sortiranje stabilan ukoliko održava relativan poredak podataka koji imaju iste vrednosti ključeva po kojima se sortiraju. Šta ovo znači je da ukoliko u početnom nizu imamo da je $x < y$ i $a[x] = a[y]$, onda element $a[x]$ treba da stoji pre $a[y]$ u konačnom sortiranom nizu. Npr. quicksort koji je objasnjen u ovoj lekciji nije stabilan, npr. prilikom sortiranja niza $3, 3, 1, 2$, trojke će zameniti mesta, dok je merge sort stabilan algoritam.

Pretpostavimo da u našem nizu celih brojeva koji treba da sortiramo svi brojevi imaju najviše d cifara. Uzmimo najpre brojeve i sortirajmo ih po poslednjoj cifri. Zatim uzmimo takav niz i sortirajmo brojeve po pretposlednjoj cifri. Sada bismo želeli da nam brojevi budu sortirani gledajući samo poslednje 2 cifre. Da bismo imali ovo, za sortiranje po određenoj cifri moramo koristiti stabilan algoritam za sortiranje. Zasto? Pretpostavimo da nemamo stabilan algoritam za sortiranje, i da posle sortiranja po poslednjoj cifri imamo brojeve 22, 23. Ukoliko algoritam za sortiranje nije stabilan, posle sortiranja po pretposlednjoj (prvoj) cifri, možemo dobiti ove brojeve u redosledu 23, 22, što nije tačno. Nastavimo ovo da radimo dokle ne dodjemo do prve cifre, posle čega ćemo imati sortiran ceo niz.

Kao stabilan algoritam za sortiranje možemo koristiti npr. mergesort ili modifikovani counting sort.

Pseudo kod radix sorta se nalazi u *Algoritam 6*.

```
================================================================================
            funkcija: radixSort
            ulaz:  a      - niz brojeva
                   d      - broj cifara brojeva u nizu a

            nakon izvršavanja funkcije radixSort(a) niz a će biti sortiran

            pretpostavimo da funkcija sort(a, i) predstavlja stabilan algoritam za
            sortiranje i sortira tako da je a[x] < a[y] ukoliko je i-ta cifra
            broja a[x] manja od i-te cifre broja a[y]
--------------------------------------------------------------------------------
      Function radixSort(a : int array, d int)
            01         For i = d downto 1 do
            02             sort(a, i)
================================================================================
```

Algoritam 6. Pseudo kod za radix sort

Složenost ovog algoritma je $O(d * slozenostSorta)$. Ukoliko npr. koristimo counting sort, složenost radix sorta je $O(d * n)$.

U praksi, ukoliko umesto baze 10, koristimo neku bazu stepena 2, npr. bazu 64, radix sort će osetno brže raditi, zbog mogućnosti baratanja bitovima (veoma brzo možemo dobiti bilo kojih 6 bitova), dok za dobijanje neke cifre moramo da delimo sa stepenima desetke, što zahteva mnogo veći broj operacija.