Selection sort

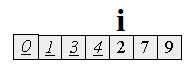
Meklējam mazāko elementu masīvā. Mainām vietām ar pirmo masīva elementu. Atkārtojam procedūru sākot ar otro masīva elementu, u.t.t.

Insertion sort

Katru i-to elementu sākot ar otro salīdzinām ar visiem iepriekšējiem skaitļiem. Salīdzināšana notiek kamēr elements kreisajā pusē nav mazāks par salīdzināmo skaitli, kad skaitlis ir atrasts, i-tais skaitlis ievietojas pa labi no atrastā skaitļa, bet visi skaitļi no (atrastā+1) līdz i-tam pārbīdās par vienu pozīciju uz labo pusi.

Ja salīdzināšanu gaitā nonācām līdz pirmajam masīva elementam tad protams jāapstājas un i-to elementu jāievieto 1-jā pozīcijā.

Piemērs:

 dati no a[0]..a[3] jau ir sakārtoti, a[4] ir i-tais elements

1. 
2. 
3. 
4. ievietošana pabeigta

Bubble sort

Jāpaņem pirmo elementu un pēc kārtas jāsalīdzina to ar visiem pārējiem. Ja salīdzināšanas rezultātā izradās ka pirmais ir lielāks, tad elementus jāsamaina vietām. Kad visi elementi ir pārbaudīti, to pašu procedūru izpildām ar visiem pārējiem elementiem pēc kārtas (2. salīdzinām ar visiem , 3. salīdzinām ar visiem, u.t.t. līdz priekšpēdējam masīva elementam ).

49762 4ar9

49762 4ar7

49762 4ar6

49762 4ar2 -> 29764

29764 9ar7 -> 27964

27964 7ar6 -> 26974

26974 6ar4 -> 24976

24976 9ar7 -> 24796

24796 7ar6 -> 24697

24697 9ar7 -> 24679

Quick sort

Pamat shēma ir sekojoša:

1. No masīva izvēlās kaut kādu elementu a[i].
2. Sākās masīva dalīšanas procedūra, kuras rezultātā visi elementi kas ir mazāki vai vienādi par a[i] – sakrājas pa kreisi no a[i], visi kas lielāki – pa labi.
3. Tagad masīvs sastāv no divām apakškopām kur kreisās apakškopas elementi ir mazāki vai vienādi par a[i], bet labas apakškopas elementi ir lielāki par a[i]. a[i] elements jau atrodas sakārtotā pozīcijā.
4. Ja kādā no apakškopām ir vairāk par 2 elementiem – rekursīvi izsaucam tai to pašu procedūru.

**Piemērs:**

masīvs 4976238

a[i]=4

labā pusē meklējam elementu kas ir mazāks vai vienāds par 4

4976238

4976238 atradām, mainām vietām

3976248

tagad kreisajā pusē sākot no nākamās pēc 4-ka pozīcijas (uz labo pusi) meklējam elementu kas ir lielāks par 4

3976248 atradām, mainām vietām

3476298

atkal labā pusē no nākamās pēc 4-ka pozīcijas (uz kreiso pusi) meklējam elementu kas ir mazāks vai vienāds par 4

3476298 atradām, mainām vietām

3276498

tagad kreisajā pusē sākot no nākamās pēc 4-ka pozīcijas (uz labo pusi) meklējam elementu kas ir lielāks par 4

3276498 atradām, mainām vietām

3246798

atkal labā pusē no nākamās pēc 4-ka pozīcijas (uz kreiso pusi) meklējam elementu kas ir mazāks vai vienāds par 4

3246798

Beidzam salīdzināšanu, jo salīdzināšanas elementu indeksi atrodas blakus pozīcijās

**Rezultāts:**

3246798 – 4 jau atrodas pareizā vietā, apakškopai „32” jāizpilda divu skaitļu salīdzināšanu, apakškopai „6798” rekursīvi jāizsauc to pašu kārtošanas funkciju.

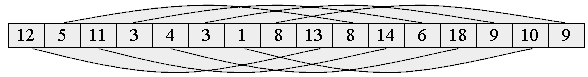
Shell sort

Tas ir insertion sort modifikācija.

Apskatīsim masīvu a[0]..a[15]



1. Sākumā ar Insertion sort jāsakarto 8 grupas no 2 elementiem (a[0],a[8]; a[1],a[9],... a[7],a[15])

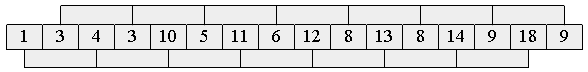


1. Tādā pašā veidā kārtojam katru grupu no 4 elementiem



Nulltā grupā būs 4,12,13,18, pirmā 3,5,8,9, u.t.t.

1. Tālāk kārtojam 2 grupas pa 8 elementiem



1. Beigās ar Insertion sort visus 16 elementus



Sagatavošanas soļi sabīda elementus uz optimālākām pozīcijām, un pēdējā solī pārvietošanu skaits būs ļoti mazs.