

Napotki za dobro programiranje

1. popolnoma definiraj problem
2. takoj začni z dokumentiranjem
3. najprej premisli, nato napiši
4. začni od zgoraj navzdol (problem postopoma razgrajuj)
5. program piši tako, da je zgrajen iz logičnih enot
6. uporablaj podprograme
7. goto stavek ni zaželen
8. uporabi imena, ki nekaj pomenijo
9. piši učinkovite komentarje
10. konstante naj bojo konstante
11. piši čitljivo in pazi na obliko
12. sintaksa naj bo pravilna
13. program preizkusi ročno
14. svoje napake popravljal sam
15. program daj v pregled še komu drugemu
16. ne boj se začeti znova od začetka

Nadalni razvoj postopkov pri razvoju programov

1. Problem razgradimo na več majših vase zaključenih podproblemov.
2. Vsak stavek v opisu postopka postopoma nadomeščamo z zaporedjem preciznejših stavkov.
3. Stavke, ki pove KAJ je treba napraviti preoblikujemo v stavke, ki povedo KAKO je treba napraviti.
4. Stavke, s katerimi opisujemo algoritem-postopek prevedemo v stavke programskega jezika
5. Stavke v programskem jeziku združimo v program in tako opis problema prevedemo v programskem jeziku.

Prednost metode razgrajevanja od vrha navzdol je v tem, da se v začetku osredotočimo na nivo, kjer je jasno, kaj je treba narediti in sistematično napredujemo proti nivoju, kjer povemo kako je treba narediti.

Zgled: urediti množico števil (urejanje)

Izberemo metodo največjega elementa – urejanje z izbiranjem

Bistvo metode je v tem, da tabelo števil pregledujemo toliko časa, dokler ne najdemo največjega števila, ki ga potem zamenjamo s številom na koncu neurejene tabele. Postopek ponavljamo od začetka do konca neurejene tabele. Novo največje število iz neurejene tabele zopet zamenjamo s zadnjim številom neurejene tabele. Tako se neurejena tabela števil krajša, urejena pa daljša. Ob koncu urejanja so števila urejena po naraščajočem vrstnem redu.

2	12	4	8	5
2	5	4	8	12
2	5	4	8	12
2	4	5	8	12
2	4	5	8	12

Problem razgradimo na zaporedje podproblemov:

-> uredi tabelo : -> a) preberi števila v tabelo
 -> b) uredi tabelo
 -> c) izpiši urejeno tabelo

b.1.) dokler je neurejena tabela ponavljaj

b.2.) poišči največje število (v neurejenem delu tabele)

b.3.) zamenjaj največje število z zadnjim številom v neurejenem delu tabele.

```
max := a[i];  
p := 1;  
for j := 2 to n do  
  if a[j] > max then  
    begin  
      p := j;  
      max := a[j];  
    end;  
end;
```

Osnovni algoritmi za urejanje

- urejanje z izbiranjem
- urejanje z vstavljanjem
- urejanje s premenami

Vsem algoritmom je skupno:

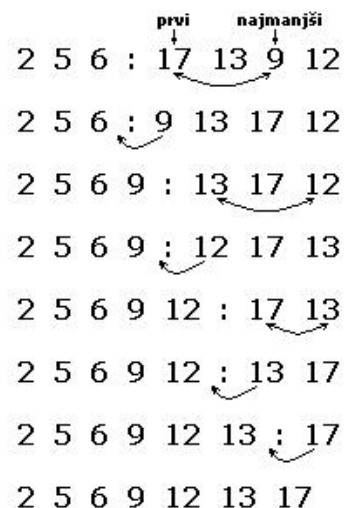
- enostavno urejanje
- preprosta zasnova
- ne najbolj učinkoviti
- primeren za majhne tabele (enostavna koda odtehta učinkovitost)

Urejanje z izbiranjem (po najmanjšem elementu)

Pri urejanju z izbiranjem na vsakem koraku izberemo v neurejenem delu tabele najmanjši element in ga zamenjamo z i-tim elementom. Če ima tabela n elementov, je potrebno ta postopek ponoviti **n-1**-krat. Zato se indeks i spreminja od 1 do n-1.

SPLOŠNO:

- poišči najmanjšega
- zamenjaj s prvim
- od 2 do n poišči najmanjšega
- zamenjaj z drugim
- imamo dva dela: UREJENI DEL / NEUREJENI DEL
- PONAVLJAJ:
- v neurejenem delu zamenjaj prvi in najmanjši element
- najmanjši (prvi) element v neurejenem delu preseli v urejen del - urejeni del se poveča za ena
- na začetku je urejeni del prazen
- končamo, ko izčrpamo neurejeni del



Povprečna časovna zahtevnost: $O(n^2)$

```
procedure uredi_z_izbiranjem(var a:tabela; n:integer);
var
  i,j:integer;
  p,max:integer;
begin
  for i := 1 to n-1 do
    begin
      max := a[i];
      p := i;
      for j := i+1 to n do
        begin
          if a[j] < max then
            begin
              p := j;
              max := a[j];
            end;
        end;
      a[p] := a[i];
      a[i] := max;
    end;
  end;
```

Urejanje z vstavljanjem

Pri urejanju z vstavljanjem vstavimo na vsakem koraku i -ti element na pravo pozicijo v urejenem delu tabele. Če ima tabela n elementov, je treba ta postopek treba ponoviti $n-1$ -krat. Zato se indeks i spreminja od 2 do n .

- Karte:
- v roki imamo nekaj urejenih kart
- vzamemo novo karto
- naredimo prostor zanjo
- vtaknemo na ustrezno mesto
- > v urejen del vstavimo nov podatek

urejeni del				:	neurejeni del						
2	5	11		:	7	*	*	*	*	*	*
2	5	7	11	:	9	*	*	*	*	*	*
...											

ALGORITEM:

- imamo dva dela: UREJENI DEL / NEUREJENI DEL
- vzemi prvi element iz neurejenega dela in ga vstavi na pravo mesto v urejeni del
- na začetku je urejeni del prvi element
- končamo, ko izčrpamo neurejeni del

Povprečna časovna zahtevnost: **$O(n^2)$**

```

procedure urejanje_vstavljanje(var x:tabela; n:integer);
var
  I,J,Pom : Integer;
begin
  for I:=2 to n do
    begin
      Pom:=X[I]; J:=I-1;
      while Pom<X[J] do
        begin
          X[J+1] := X[J];
          dec(J);
        end;
      X[J+1] := Pom;
    end;
  end;
end;

```

Urejanje s premenami (mehurčno urejanje)

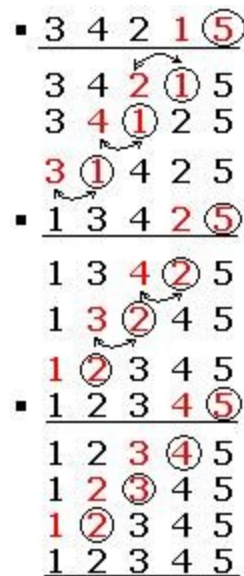
Pri urejanju s premenami na vsakem koraku premaknemo en element na pravo mesto, ostale pa pogreznemo eno ali več mest bliže končnemu. Postopek ponovimo **n-1**-krat.

ALGORITEM:

- vzamemo zadnji element in ga primerjamo s svojim predhodnikom:
- če je predhodnik manjši, potem sta elementa v pravem vrstem redu in zgrabimo predhodnika
- če predhodnik ni manjši, ju zamenjamo
- ...
- najlažji element splava na površje
- plavanje - zamenjave

Časovna zahtevnost: $O(n^2)$

```
procedure bubble_sort(var x: tabela; n: integer);  
var I, J, Pom: Integer;  
begin  
  for I:=2 to n do  
    for J:=n downto I do  
      begin  
        if X[J-1]>X[J] then  
          begin  
            Pom:=X[J-1];  
            X[J-1]:=X[J];  
            X[J]:=Pom;  
          end;  
        end;  
      end;  
end;
```



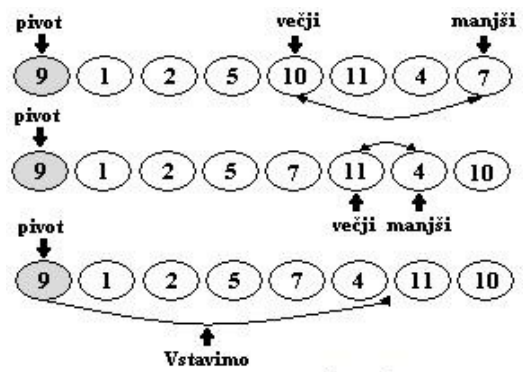
Hitro urejanje

ALGORITEM:

- izberi si delilni element (pivot)
- razdeli na dva dela (\leq pivot, \geq pivot)
- z istim postopkom uredi oba dela
-

```

procedure qsort(var a: tabela; spodnji,
zgornji: integer);
var
  levi, desni, pivot, X: integer;
begin
  pivot:=a[(spodnji+zgornji) div 2];
  levi:=spodnji;
  desni:=zgornji;
  while levi<=desni do
  begin
    while a[levi] < pivot do
      levi:=levi+1;
    while a[desni] > pivot do
      desni:=desni-1;
    if levi<=desni then
    begin
      X := a[levi];
      a[levi] := a[desni];
      a[desni] := X;
      levi:=levi+1;
      desni:=desni-1;
    end;
  end;
  if levi>spodnji then
    qsort(a, spodnji,desni); { Sortiramo levo
    stran }
  if zgornji>desni then
    qsort(a, levi ,zgornji); { Sortiramo desno
    stran }
end;
  
```



uredimo levo in desno od pivota

