Podatkovna struktura kopica

↓↓↓
Andrej Taranenko

→ **Tokrat bomo predstavili podatkovno strukturo, imenovano kopica. V literaturi se pojavlja več različnih definicij strukture, ki se skriva pod imenom kopica. V našem prispevku bo kopica dvojiško drevo z zahtevanimi dodatnimi lastnostmi, podrobneje definirana v nadaljevanju. Ponovimo na začetku nekaj o dvojiških drevesih.**

Dvojiško drevo in predstavitev s poljem

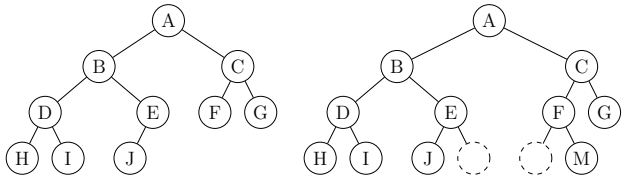
Dvojiško drevo je podatkovna struktura, ki je bodisi prazna bodisi za vsako vozlišče velja, da ima največ dva otroka.

Z besedo *vozlišče* imamo v mislih strukturo, ki vsebuje vrednost, imenovano *ključ*, ter dve vozlišči, imenovani *levi* in *desni otrok*. Kadar govorimo samo o otrocih, bomo v mislih imeli poljubnega izmed njiju ali oba. Vozlišče v je *starš* vozlišča u, če je u otrok vozlišča v.

V dvojiškem drevesu, ki ni prazno, imamo posebno vozlišče, imenovano *koren* oz. *korensko vozlišče*. Korensko vozlišče imamo za prvi *nivo* drevesa. Njegovi otroci so na drugem nivoju drevesa, njihovi otroci na tretjem itd. Naj bo v vozlišče dvojiškega drevesa. Če pogledamo drevo, katerega korensko vozlišče je levi otrok vozlišča v, govorimo o *levem poddrevesu*, drevo, katerega koren je desni otrok vozlišča v, pa je *desno poddrevo*. Vozlišče drevesa, ki nima otrok, imenujemo *list*.

Na sliki 1 vidimo primera dveh dvojiških dreves. V obeh je vozlišče s ključem A korensko vozlišče. Če se omejimo na levo drevo na sliki 1, je drevo, ki ga tvorijo vozlišča s ključi C, F in G, desno poddrevo vozlišča s ključem A. Listi drevesa pa so vozlišča s ključi F, G, H, I in J. Drevo ima štiri nivoje. Vozlišče s ključem A je na prvem nivoju drevesa, vozlišči s ključema B in C na drugem nivoju drevesa, na tretjem so vozlišča s ključi D, E, F in G, na zadnjem (četrtem) nivoju pa vozlišča s ključi H, I in J.

Za dvojiško drevo, ki je *levo poravnano*, velja, da ima na vseh nivojih, razen morda zadnjem, vsa možna vozlišča. Na zadnjem nivoju drevesa pa dopuščamo, da na desni strani manjkajo vozlišča. Primer levo poravnanega drevesa vidimo na levi strani slike 1; primer devesa, ki ni levo poravnano, saj med vozliščem J in vozliščem M manjkajo črtkasto narisana vozlišča, pa vidimo na desni strani slike 1.



SLIKA 1.
Levo poravnano dvojiško drevo in dvojiško drevo, ki ni levo poravnano.

Dvojiško drevo lahko v računalniku med drugim predstavimo s poljem tako, da ključe v vozliščih od zgoraj navzdol in od leve proti desni za povrstjo shranimo v polje. Levo drevo na sliki 1 bi na ta način predstavili s poljem v tabeli 1.

Če ne bi zahtevali, da je drevo levo poravnano, bi pri tej predstavitvi v polju dobili prazna (neizkoriščena) mesta. V tabeli 2 vidimo desno drevo s slike 1 predstavljeno s poljem.

indeks	0	1	2	3	4	5	6	7	8	9
vrednost	A	B	C	D	E	F	G	H	I	J

TABELA 1.
Predstavitev dvojiškega drevesa s poljem

indeks	0	1	2	3	4	5	6	7	8	9	10	11	12
vrednost	A	B	C	D	E	F	G	H	I	J			M

TABELA 2.
Dvojiško drevo, ki ni levo poravnano, predstavljeno s poljem.

Predpostavimo, da so elementi polja indeksirani od 0 naprej. Sami lahko preverite, da ima vozlišče, ki ga v polje shranimo na indeks i, levega otroka shranjenega na indeksu $2i + 1$, desnega pa na indeksu $2i + 2$. Starš vozlišča, shranjenega na indeksu i, je shranjen na indeksu $\lfloor \frac{i-1}{2} \rfloor$. V vseh primerih zapisano seveda velja, če element z ustreznim indeksom v polju sploh obstaja.

Kopica in operacije na njej

Kopica je levo poravnano dvojiško drevo, v katerem za vsako vozlišče velja: Če je A starš vozlišča B, potem je ključ v vozlišču A večji ali enak ključu v vozlišču B. Če vozlišče nima otrok, velja, da je lastnosti zadoščeno.

Tako definirana kopica se imenuje tudi *maksimalna kopica*. Podobno bi definirali *minimalno kopico*, kjer bi za vsako vozlišče veljalo, da je ključ v vozlišču manjši od ključa v otroku. V nadaljevanju bomo maksimalni kopici rekli samo kopica. Ker bomo kopico predstavili s poljem, zahtevamo, da imamo levo poravnano dvojiško drevo.

Primer kopice vidimo na sliki 2. Opazimo lahko, da je v korenskem vozlišču kopice vedno ključ največje vrednosti, kar je tudi splošna lastnost kopice.

Spoji kopici

Prvi postopek nad kopicami, ki ga bomo predstavili, se imenuje *spoji kopici*. Gre za naslednje: podani imamo dve kopici in še eno vozlišče. Vse troje bi radi spojili (združili) v eno dvojiško drevo, ki bo tudi kopica.

Začetno stanje problema je predstavljeno na sliki 3, na kateri vidimo dve kopici (rdeče in zeleno obarvano drevo) ter dodatno (modro obarvano) vozlišče, ki bi ga želeli povezati s podanima kopicama. Rdeče in zeleno drevo zadoščata lastnostim kopice, celotno drevo pa ne predstavlja kopice, saj modro vozlišče nima ključa, večjega od desnega otroka.

Postopek spoji kopici izvedemo tako: naše trenutno vozlišče na začetku postane dodatno (modro) vozlišče. Dokler s trenutnim vozliščem nismo v listu kopice ali pa trenutno vozlišče krši lastnost kopice (ključ v trenutnem vozlišču je manjši od vsaj enega izmed ključev v otrocih), zamenjaj ključ s ključem večjega izmed otrok in se prestavi na vozlišče, s katerim smo zamenjali vrednosti.

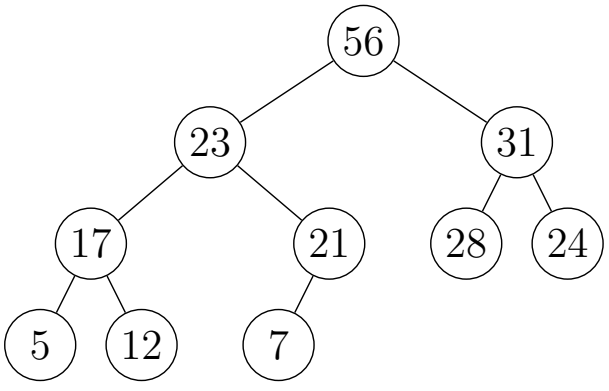
Korake postopka spoji kopici za primer s slike 3 vidimo na sliki 4, pri čemer je trenutno vozlišče vedno obarvano z modro.

Zgradi kopico

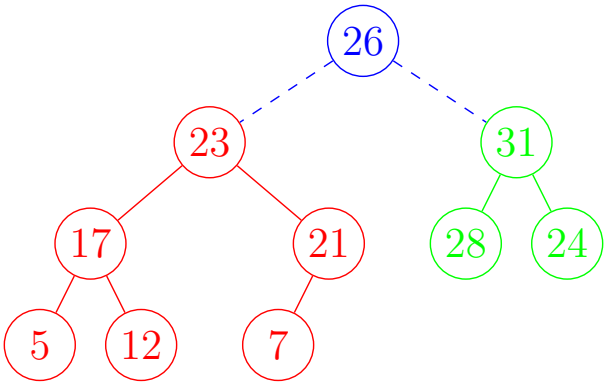
Pri postopku *zgradi kopico* predpostavimo, da imamo podano polje, katerega elementi ne zadoščajo nujno lastnostim kopice. Elemente polja želimo preurediti tako, da bo celotno polje predstavljalo kopico.

Poglejmo si primer, kjer je začetno stanje predstavljeno s sliko 5.

Pri izgradnji kopice si bomo pomagali s postopkom *spoji kopici*. Ker mora vsako vozlišče drevesa

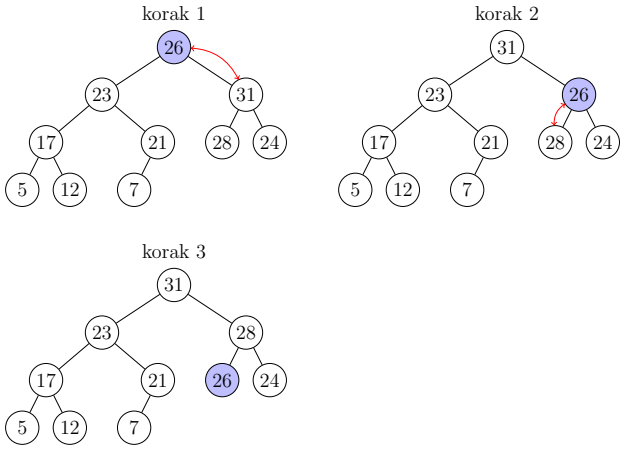


SLIKA 2.
Primer kopice



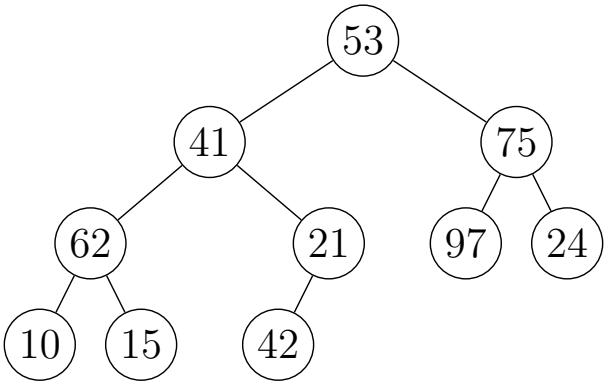
SLIKA 3.
Začetno stanje postopka spoji kopici

zadoščati lastnosti kopice, bomo po vrsti od desne proti levi in od spodaj navzgor spajali posamezne kopice. Listi dvojiškega drevesa vedno zadoščajo lastnosti kopice, saj nimajo otrok. Torej prvo vozlišče, pri katerem bomo izvedli postopek spoji kopici, bo starš najbolj desnega lista na zadnjem nivoju drevesa, zadnje bo pa korensko vozlišče.



SLIKA 4.
Koraki postopka spoji kopici

indeks	0	1	2	3	4	5	6	7	8	9
vrednost	53	41	75	62	21	97	24	10	15	42



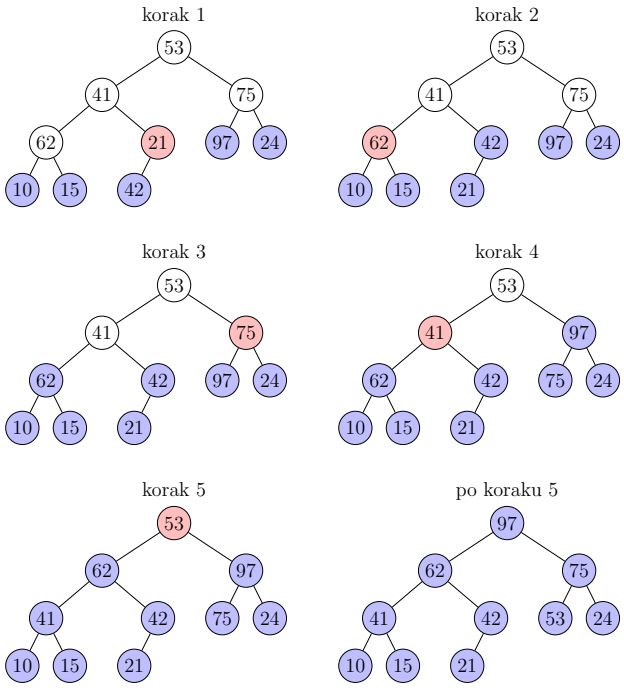
SLIKA 5.
Polje, ki ne predstavlja nujno kopice, in pripadajoče dvojiško drevo.

Na sliki 6 vidimo posamezne korake (en klic postopka spoji kopici) izvajanja postopka zgradi kopico. Poddrevesa z modro obarvanimi vozlišči že zadoščajo lastnostim kopice. Vozlišče obarvano z rdeče pa na vsakem koraku predstavlja vozlišče, ki ga spajamo s pripadajočima poddrevesoma (kopicama). Po zadnjem koraku dobimo dvojiško drevo, ki predstavlja kopico.

Odstrani največji element iz kopice

Tokrat želimo iz obstoječe kopice odstraniti največji element. V mislih imejmo, da bo v računalniku kopica shranjena v polju, tako da ne bomo elementa dejansko odstranili, ampak se bomo pretvarjali, da je velikost kopice (število elementov v kopici) manjša od števila vseh elementov v polju.

Največji element kopice učinkovito odstranimo tako: Zamenjaj največji element kopice (korenski) z najbolj desnim listom zadnjega nivoja. Zmanjšaj velikost kopice za en element. Spoji »novi« korenski element s kopicama, ki ju predstavljata njegova otroka.

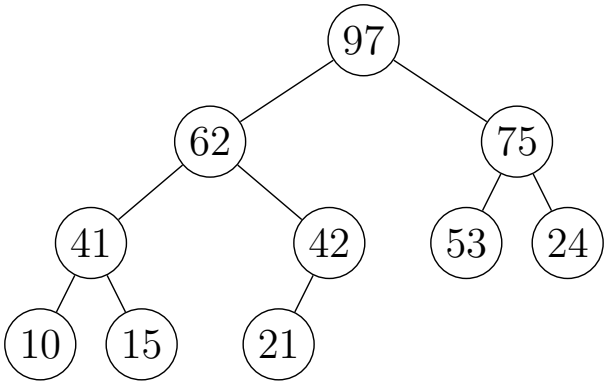


SLIKA 6.
Koraki postopka zgradi kopico

Poglejmo primer odstranjevanja največjega elementa iz kopice s slike 7.

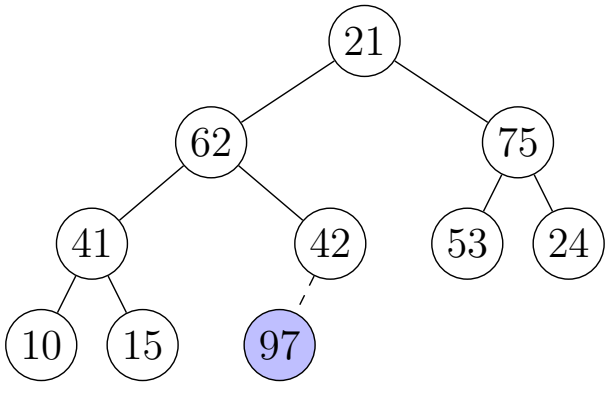
Iz kopice želimo odstraniti največji element. Na prvem koraku ga zamenjamo z najbolj desnim listom zadnjega nivoja in zmanjšamo velikost kopice za 1. Na slikah 8 in 9 to pomeni, da si predstavljamo, da modro obarvano vozlišče (element polja) ni več del kopice.

indeks	0	1	2	3	4	5	6	7	8	9
vrednost	97	62	75	41	42	53	24	10	15	21



SLIKA 7.
Kopica in pripadajoče polje

indeks	0	1	2	3	4	5	6	7	8	9
vrednost	21	62	75	41	42	53	24	10	15	97



SLIKA 8.
Prvi korak odstranjevanja največjega elementa

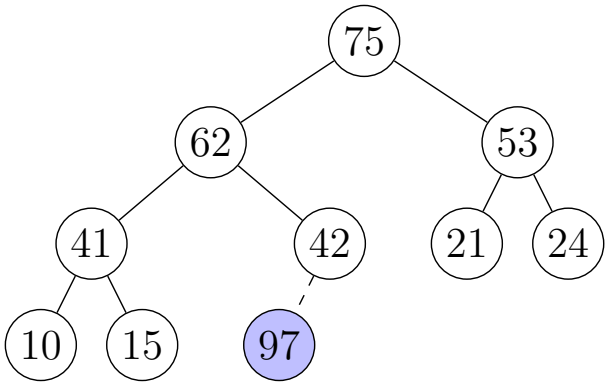
Ker smo spremenili prvi element drevesa, se lahko zgodi, da krši pravilo kopice. Drugih elementov, ki so ostali v kopici, nismo spreminjali, zato so ohranili lastnost kopice. Imamo torej dve kopici (levo in desno poddrevo) korenskega vozlišča in korensko vozlišče, ki morebiti krši lastnost kopice. Izpolnjeni so torej pogoji, da izvedemo postopek spoji kopici. Ne pozabite, modro obarvano vozlišče ni več del kopice. Po izvedenem postopku spajanja korenskega vozlišča s kopicama v levem in desnem poddrevesu dobimo kopico predstavljeno z belimi vozlišči na sliki 9.

Uporaba kopice

Kopica je zaradi svojih lastnostih koristna pri različnih problemih. Ker imamo v kopici v korenskem vozlišču na vsakem koraku vedno element z največjim ključem, je kopica zelo primerna za uporabo kot *prednostna vrsta*. V prednostno vrsto namreč dajemo elemente v nekem vrstnem redu, iz nje jih pa pridobivamo v vrstnem redu glede na njihovo prioriteto, ki pri implementaciji s kopico, predstavlja ključe, glede na katere gledamo urejenost. Kot smo povedali v prejšnjih razdelkih, imamo v kopici v korenu vedno največji element, ki ga znamo enostavno odstraniti.

Drugi primer uporabe, ki ga bomo predstavili tukaj, pa je urejanje podatkov v polju. Problem je sle-

indeks	0	1	2	3	4	5	6	7	8	9
vrednost	75	62	53	41	42	21	24	10	15	97



SLIKA 9.
Dobljena kopica po odstranjevanju največjega elementa

deč: imamo polje napolnjeno s podatki. Podatke v polju želimo urediti nepadajoče.

Tokrat bomo problem urejanja podatkov rešili s kopico. Pri odstranjevanju največjega elementa iz kopice lahko opazimo, da na mesto zadnjega elementa kopice (preden spremenimo velikost kopice) dobimo ravno največji element. Če torej iz dobljenih kopic zaporedoma odstranjujemo največji element, dobimo v polju ključe, urejene v nepadajočem vrstnem redu. Elemente odstranjujemo, dokler ima dobljena kopica več kot en element. Seveda moramo pred odstranjevanjem ključev iz podatkov polja zgraditi začetno kopico. Temu postopku urejanja pravimo *urejanje s kopico*. Urejanje s kopico ima časovno zahtevnost $O(n \log n)$ in je zato po hitrosti primerljivo z najhitrejšimi algoritmi urejanja.

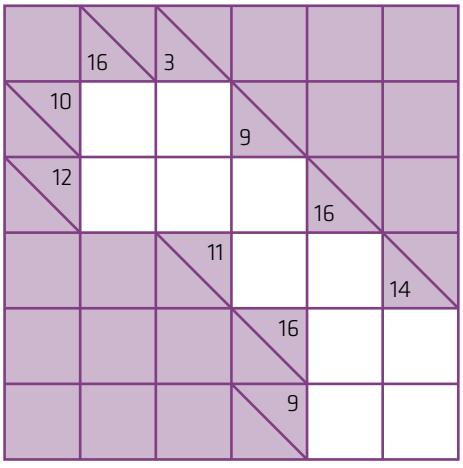
Poglejmo postopek urejanja s kopico na spodnjem primeru. Tokrat bomo zapisovali le polja, ki jih dobimo, in ne bomo izrisovali pripadajočih dvojiških dreves. Postopek je prikazan na sliki 10, pri čemer znova velja, da modro obarvani elementi niso več del kopice.

indeks	0	1	2	3	4	5	6	7	8	9
začetno stanje polja	53	41	75	62	21	97	24	10	15	42
izvedemo *zgradi kopico*	97	62	75	41	42	53	24	10	15	21
odstrani največjega	75	62	53	41	42	21	24	10	15	97
odstrani največjega	62	42	53	41	15	21	24	10	75	97
odstrani največjega	53	42	24	41	15	21	10	62	75	97
odstrani največjega	42	41	24	10	15	21	53	62	75	97
odstrani največjega	41	21	24	10	15	42	53	62	75	97
odstrani največjega	24	21	15	10	41	42	53	62	75	97
odstrani največjega	21	10	15	24	41	42	53	62	75	97
odstrani največjega	15	10	21	24	41	42	53	62	75	97
odstrani največjega	10	15	21	24	41	42	53	62	75	97
postopek zaključen	10	15	21	24	41	42	53	62	75	97

SLIKA 10.
Koraki postopka uredi s kopico na primeru podanega polja

Križne vsote

→ Naloga reševalca je, da izpolni bele kvadratke s števkami od 1 do 9 tako, da bo vsota števk v zaporednih belih kvadratkih po vrsticah in po stolpcih enaka številu, ki je zapisano v sivem kvadratku na začetku vrstice (stolpca) nad (pod) diagonalo. Pri tem morajo biti vse števke v posamezni vrstici (stolpcu) različne.



REŠITEV KRIŽNE VSOTE