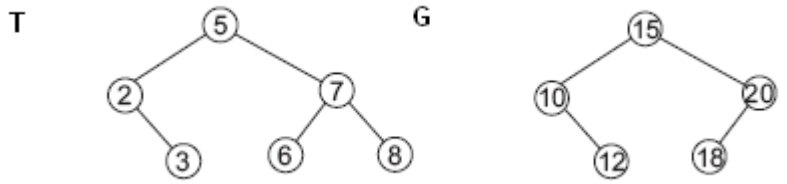
V čas

1. Konkatenacija je operacija nad dva skupa koja zadovoljavaju uslov da su svi ključevi u jednom skupu manji od svih ključeva u drugom skupu. Rezultat konkatenacije je unija skupova. Primeniti algoritam sa prethodnog časa na data dva binarna stabla pretrage. Vremenska složenost algoritma mora biti $O(h)$ u najgorem slučaju, gde je h veća od visina dva stabla.



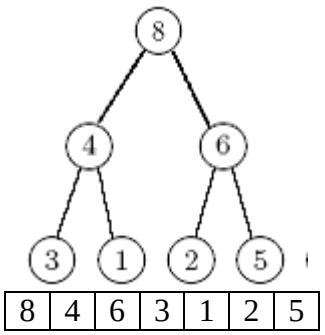
5.1 Hip (eng. *heap*)

Hip je binarno stablo koje zadovoljava uslov hipa: ključ svakog čvora veći je ili jednak od ključeva njegovih sinova.

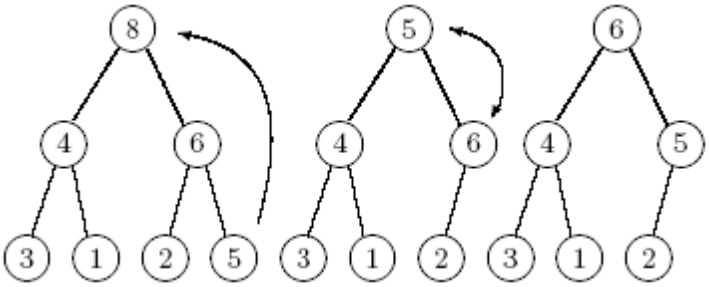
Hip se može realizovati implicitno i eksplicitno.
Mi ćemo se baviti kao kod profesora u knjizi **implicitnim** realizacijama hipa. Dakle, ako hip ima n elemenata, onda se za smeštanje elemenata koriste lokacije u nizu A sa indeksima A[1..n], tako da ako element indeksa ***i*** predstavlja čvor stabla, tada
- element indeksa ***2*i*** predstavlja levo dete čvora,
- element indeksa ***2*i+1*** predstavlja desno dete čvora.
Na primer, deca čvora A[1] su elementi A[2], A[3].

Podsetite se algoritma iz profesorove knjige **Skini_max_sa_hipa (A,n)** za uklanjanje najvećeg elementa sa hipa A dimenzije n.

Neka je dat hip.



Treba ukloniti ključ A[1], a potom transformisati niz A tako da opet zadovoljava svojstvo hipa. Najzgodnije je u A[1] zapisati krajnji element niza i smanjiti veličinu hipa. Problem je što tada možda nije zadovoljeno pravilo hipa.



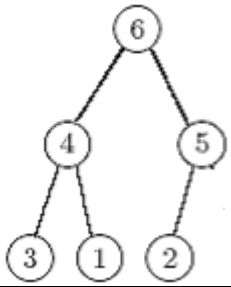
Dovođenje niza u stanje koje zadovoljava pravilo hipa vrši se prema sledećem algoritmu:

Algoritam: uređenje hipa kada vrh hipa nije u skladu s pravilom hipa
1. Započni s indeksom koji predstavlja vrh hipa i smesti taj element u promenljivu x. Nadalje se uzima da je vrh hipa "prazan".
2. Analiziraj decu praznog čvora i odredi koje dete je vece.
3. Ako je ključ od x veci od ključa veceg deteta smesti x u prazan čvor i završi, inače upiši element veceg deteta u prazan čvor i postavi da čvor veceg deteta bude prazan čvor. Ponovi korak 2.

Rezultat je hip:

6	**4**	**5**	**3**	**1**	**2**

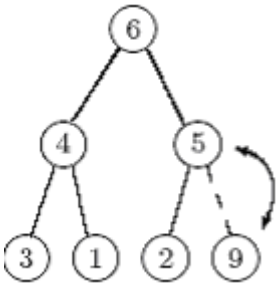
Podsetite se algoritma iz profesorove knjige **Upis_u_hip (A,n,x)** za upis elementa x u hip A dimenzije n.
Neka je dat hip



6	4	5	3	1	2

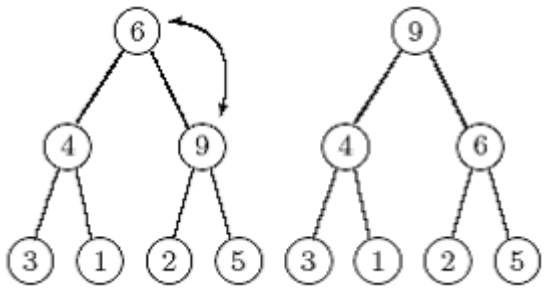
i neka treba ubaciti ključ 9.

Ako želimo dodati element u prioritetni red možemo ga dodati na kraj niza A. To je ekvivalentno dodavanju krajnjeg desnog lista u stablu. Ta operacija ima za posledicu da se N uveća za jedan, ali i da možda stablo ne zadovoljava pravilo hipa.



Da bi se zadovoljilo pravilo hipa koristi se postupak opisan u sledećem algoritmu:

Algoritam: Umetanje
1. Element iza krajnjeg elementa niza se tretira kao prazan (to je sledeći krajnji desni list stabla). Podrazumeva se da je kapacitet niza veći od broja elemenata u nizu.
2. Ako je ključ roditelja praznog čvora veći od ključa elementa x, tada se element x upisuje u prazni čvor. Time je postupak završen.
3. Ako je ključ roditelja praznog čvora manji od ključa elementa x, tada se element roditelja kopira u prazni čvor, a čvor u kojem je bio roditelj se uzima kao prazni čvor. Postupak se ponavlja korakom 2.



Rezultat je hip:

9	**4**	**6**	**3**	**1**	**2**	**5**

Primene hip-a

Lista sa prioritetom: dinamički skup čiji elementi se uklanjaju po redosledu veličina, počev od najvećeg. Na primer, objekat sa najvećim prioritetom se nalazi u korenu heap-a i trivijalno se uzima iz strukture. Ali ako se ukloni koren, tada ostaju dva podstabla i moraju se *efikasno* spojiti u jedno stablo koje će ponovo imati ***heap*** svojstvo.
Korist od upotrebe heap strukture zasniva se na svojstvu da se moze izvaditi objekat najvišeg prioriteta i ubaciti za **O(log*n*)** vremena.

Heap Sort: od kolekcije elemenata kreira se heap za (O(n)) vremena i potom se uklanjaju elementi iz heap-a, a svako uklanjanje ima složenost proporcionalnu visini heap-a, tj. (O(log n)), te ukupna složenost je (O(n log n)).

1. Odrediti izgled implicitno predstavljenog hipa koji se dobija umetanjem redom brojeva 5, 1, 3, 9, 6, 4, 7, 8 polazeći od praznog hipa. Prikazati izgradnju hipa tabelom, čiji svaki red odgovara jednoj promeni hipa. Zatim odrediti izgled hipa dobijenog uklanjanjem najvećeg elementa.

Korak 1 – dodaj 5

5	

Korak 2 - dodaj 1

5	1

Korak 3: - dodaj 3

5	1	3

Korak 4 - dodaj 9 i dva podkoraka preuređivanja hipa u I i III redu:

5	**1**	3	**9**
5	9	3	1
5	**9**	3	1
9	5	3	1

Korak 5 – dodaj 6 i jedan podkorak preuređivanja hipa

9	**5**	3	1	**6**
9	6	3	1	5

Korak 6 – dodaj 4 i jedan podkorak preuređivanja hipa

9	6	**3**	1	5	**4**
9	6	4	1	5	3

Korak 7 – dodaj 7 i jedan podkorak preuređivanja hipa

9	6	**4**	1	5	3	**7**
9	6	7	1	5	3	4

Korak 7 – dodaj 8 i dva podkoraka preuređivanja hipa u I i III redu:

9	6	7	**1**	5	3	4	**8**
9	6	7	8	5	3	4	1
9	**6**	7	**8**	5	3	4	1
9	8	7	6	5	3	4	1

Uklanjanje max elementa 9

Korak 1 - ukloni 9

8	7	6	5	3	4	1

Korak 2 – kopiraj poslednji član tj. 1 u koren hipa

1	8	7	6	5	3	4

Korak 3 – preuređivanje hipa, max (8,7) zameni sa 1

1	**8**	7	6	5	3	4
8	1	7	6	5	3	4

Korak 4 – preuređivanje hipa, max (6,5) zameni sa 1

8	**1**	7	**6**	5	3	4
8	6	7	1	5	3	4

2. Neka je *A* vektor koji služi za implicitno predstavljanje hipa. Koliki najmanji broj elemenata hipa može da zauzme niz dužine 16?

REŠENJE: Kako je zauzeto polje na poziciji 16 => mora biti zauzeto i polje na poziciji 8 za čuvanje oca čvora 16. Zato što po definiciji implicitne reprezentacije hipa preko niza, element A[16] odgovara čvoru čiji predak je čvor pridružen elementu A[8]. Sličnim rezonom, dolazi se do zaključka da moraju biti zauzete za pretke i pozicije 4, 2, 1. Dakle, minimalan broj elemenata je 5. ($\log_2 16 + 1$)

3. Konstruisati algoritam za formiranje hipa koji sadrži sve elemente dva hipa veličine *n* i *m*. Hipovi su predstavljeni eksplicitno (svaki čvor ima pokazivače na svoja dva sina). Vremenska složenost treba da bude u najgorem slučaju $O(\log(m + n))$.

REŠENJE: K1. uklanja se proizvoljan element iz nekog hipa (npr. veći koren) i izvedu se popravke hipa na način koji je opisan u sekciji uklanjanja elementa iz hipa

K2. izabrani element se umeće kao koren novog hipa, tako da njegovi sinovi budu koreni dva stara hipa

Popravka dobijenog hipa (stabla) uz eventualno premeštanje naniže elementa iz korena zahteva O(log(m+n)) koraka zbog dimenzije novog hipa.

4. Primeniti algoritam iz prethodnog zadatka na dva data hipa.