Kontejner hip **<queue>**

Ko ovo ne zna, kečina.

Deklaracija: **priority\_queue <tip\_podatka> ime\_hipa;**

Hip je kontejner / struktura podataka prvenstveno namenjena smeštanju podataka koji imaju sledeće osobine :

1. Hip je binarno stablo, ali u kome je moguće pristupiti samo **najvećem** članu kontejnera.
2. Članovi se dodaju u hip, (kontejner će ih smestiti kako treba) u logN vremenu.
3. Članovi se skidaju isključivo sa vrha hipa nakon čega sledeći najveći dolazi na vrh.

Hip je moguće implementirati i uz pomoć sopstvenog niza. Za nijansu je više koda koji nije ni jednostavan, ali se dobija na brzini (empirijski oko 50%).

**priority\_queue** **<tip> Q;**

Ako je potrebno saznati koliko članova ima u hip-u sledeća funkcija to daje: **Q**.**size**();

Dodavanje člana u hip: **Q**.**push**(vrednost);

Skidanje člana iz hip-a (sa vrha): **Q**.**pop**();

Pristup vrhu hipa: **Q**.**top**();

Ako je potrebno pražnjenje hipa u toku izvršavanja programa to mora peške odpopovati;  
  
To i ostale metode i funkcije vezane za hip kontejner imate na [**tutorijalu za C++ ←(click here)**](http://www.cplusplus.com/reference/queue/priority_queue/)

Ako je potreban minimalni hip, ili neki drugi ili dvostruki kriterijum, tada se operatorom < u strukturi sloga, to mora definisati primer: hoćemo da nam na vrhu heap-a bude najmanja tacka po x osi, a ako ih ima vise, onda najmanja po y.

struct **slog**{

int x,y;

bool operator < (const slog &a) const {

return (x>a.x) or (x==a.x and y>a.y));} // ako zelite maximum na vrhu hipa, onda je znak < (po defaultu)

};

**priority\_queue** **<slog> Q;**

Zadaci:

1. Code Chef - [Table Sum](https://www.codechef.com/IOIPRAC/problems/INOI1202)
2. Code Chef - [Chef and Way](https://www.codechef.com/problems/CHRL4)

1. Code Chef - [Polinomials](https://www.codechef.com/problems/POLY)