**STRUCTURE**

typedef struct Char\_s

{

char c;

int freq;

Char\_s(char C,int Freq) // Constructor

{

c=C;

freq=Freq;

}

} Char;

// Overriding > operator

bool operator> (const Char &l, const Char &r)

{

return (l.freq)>(r.freq);

}

// Overriding < operator

bool operator< (const Char &l, const Char &r)

{

return (l.freq)<(r.freq);

}

// Overriding << operator… Now you can print like (cout << o << endl; //where o is object)

ostream& operator<< (ostream &os, const Char &o)

{

os << o.c << " : " << o.freq << endl;

return os;

}

**PRIORITY QUEUE**

priority\_queue<int> Q;

* This will initialize priority queue as max-heap default.

priority\_queue<int, vector<int>, greater<int> > Q; // greater is inbuilt function.

* This will initialize priority queue as **min-heap**.

priority\_queue<Char, vector<Char>, **compare** > Q; // Char is structure (int a,int b)

* First of all, you have to override > and < operators and implement. (Check structure section)
* Below compare will implement min-heap

struct compare

{

bool operator()(const Char& l, const Char& r)

{

return l > r;

}

};

Below compare will implement max-heap

struct compare

{

bool operator()(const Char& l, const Char& r)

{

return l < r;

}

};

**SORTING**