Venice Set

1. Imagine you have an empty land and the government can make queries of the following type:
    1. Make a building with A floors.
    2. Remove a building with B floors.
    3. Remove C floors from all the buildings.
    4. (A lot of buildings can be vanished)
    5. Which is the smallest standing building.
    6. (Obviously buildings which are already banished dont count)
2. The operations 1,2 and 4 seems very easy with a set, but the 3 is very cost effective probably O(N) so you might need a lot of workers.
3. But what if instead of removing C floors we just fill the streets with enough water (as in venice) to cover up the first C floors of all the buildings :
4. Well that seems like cheating but at least those floor are now vanished 😃. So in order to do that we apart from the SET we can maintain a global variable which is the water level.
5. So in fact if we have an element and want to know the number of floors it has we can just do height - water_level and in fact after water level is for example 80.
6. If we want to make a building of 3 floors we must make it of 83 floors so that it can tough the land. So the implementation of this technique might look like this.

```cpp
struct veniceset
{
    multiset<ll> s;
    ll water_level = 0;

    void add(ll v)
    {
        s.insert(v + water_level);
    }
    void remove(ll v)
    {
        s.erase(s.find(v + water_level));
    }
    void updall(ll v)
    {
        water_level += v;
    }
    ll getmin()
    {
        return *s.begin() - water_level;
    }
    ll size()
    {
        return s.size();
    }
};
```