按排名分裂、合并的堆搜索树 FHQ Treap

`std::cin` 不可以读取字符

要用 `std::cin.get(char)` 来读取字符

```
#include <bits/stdc++.h>


const long long N = 2e6;
long long n;
long long tr[N], key[N], size[N], ls[N], rs[N], ant;
long long ltr, rtr; // 以光标为分割线

void set_size(long long u)
{
    if (u == 0) return;
    size[u] = 1 + size[ls[u]] + size[rs[u]];
}

void split(long long u, long long k, long long& l, long long& r)
{
    if (u == 0)
    {
        l = r = 0;
        return;
    }

    long long t = size[ls[u]] + 1;

    if (t == k)
    {
        l = u;
        r = rs[u];
        rs[u] = 0;
    }
    else if (k > t)
    {
        l = u;
        split(rs[u], k - t, rs[u], r);
    }
    else
    {
        r = u;
        split(ls[u], k, l, ls[u]);
    }

    set_size(l);
    set_size(r);
}
```

```
void meld(long long& u, long long l, long long r)
{
    if (l == 0 || r == 0)
    {
        u = l + r;
        set_size(u);
        return;
    }

    if (key[l] > key[r])
    {
        u = l;
        meld(rs[u], rs[u], r);
    }
    else
    {
        u = r;
        meld(ls[u], l, ls[u]);
    }

    set_size(u);
}

void insert(long long x)
{
    long long t = ++ ant;
    tr[t] = x;
    key[t] = rand();
    meld(rtr, t, rtr);
}

void move(long long k)
{
    long long root;
    meld(root, ltr, rtr);
    // if (k == 0)
    // {

    // }
    split(root, k, ltr, rtr);
}

void del(long long len)
{
    long long t;
    split(rtr, len, t, rtr);
}

void dfs(long long u)
{
    if (u == 0) return;
    dfs(ls[u]);
    std::cout << (char)tr[u];
```

```
    dfs(rs[u]);
}

void get(long long len)
{
    long long t;
    split(rtr, len, t, rtr);
    dfs(t);
    std::cout << '\n';
    meld(rtr, t, rtr);
}

void prev()
{
    long long t;
    split(ltr, size[ltr] - 1, ltr, t);
    meld(rtr, t, rtr);
}

void next()
{
    long long t;
    split(rtr, 1, t, rtr);
    meld(ltr, ltr, t);
}

std::string gss(long long len)
{
    long long l = 32, r = 126;
    std::string res;
    while ((long long)res.size() < len)
    {
        char c; std::cin.get(c);
        if (!((long long) c >= l && (long long) c <= r)) continue;
        res += c;
    }
    return res;
}

void solve()
{

    std::cin >> n;
    std::string s1, s2;
    long long x;
    while (n --)
    {
        std::cin >> s1;
        if (s1 == "Move")
        {
            std::cin >> x;
            move(x);
        }
        else if (s1 == "Insert")
```

```
        {
            std::cin >> x;
            s2 = gss(x);
            for (long long i = x - 1; ~i; i --)
            {
                insert(s2[i]);
            }
        }
        else if (s1 == "Delete")
        {
            std::cin >> x;
            del(x);
        }
        else if (s1 == "Get")
        {
            std::cin >> x;
            get(x);
        }
        else if (s1 == "Prev") prev();
        else if (s1 == "Next") next();
    }

}

int main()
{
    std::ios::sync_with_stdio(0);
    std::cin.tie(0); std::cout.tie(0);

    solve();
    return 0;
}
```