

```
1 // 遅延(add) skew heap (最小値 heap)
2 // verified: http://judge.u-aizu.ac.jp/onlinejudge/review.jsp?rid=3355743
3 // new をやめると少し早くなる
4 template <typename T>
5 class LazySkewHeap
6 {
7     struct Node
8     {
9         T val, add;
10         Node *l, *r;
11
12         Node(T v) : val(v), add(0), l(NULL), r(NULL) {}
13         Node() {}
14     };
15
16     Node *root;
17
18 private:
19     void lazy(Node *a)
20     {
21         if (a->l)
22             a->l->add += a->add;
23         if (a->r)
24             a->r->add += a->add;
25         a->val += a->add;
26         a->add = 0;
27     }
28
29     Node *meld(Node *a, Node *b)
30     {
31         if (!a)
32             return b;
33         if (!b)
34             return a;
35         // min Heap
36         if (a->val + a->add > b->val + b->add)
37             swap(a, b);
38         lazy(a);
39         a->r = meld(a->r, b);
40         swap(a->l, a->r);
41         return a;
42     }
43
44 public:
45     LazySkewHeap() : root(NULL) {}
46
47     void meld(LazySkewHeap *lsh)
48     {
49         root = meld(root, lsh->root);
50     }
51
52     void push(T v)
53     {
54         root = meld(root, new Node(v));
55     }
56
57     void add(T add)
58     {
59         assert(root != NULL);
60         root->add += add;
61     }
62
63     T top()
64     {
65         assert(root != NULL);
66         return root->val + root->add;
67     }
68
69     void pop()
70     {
71         assert(root != NULL);
72         lazy(root);
73         root = meld(root->l, root->r);
74     }
75
76     bool empty()
77     {
78         return !root;
79     }
80 };
81
```