KTH Royal Institute of Technology

Omogen Heap

Simon Lindholm, Johan Sannemo, Mårten Wiman

2020-08-12

Contest (1)

template.cpp

9 lines

```
#include <bits/stdc++.h>
using namespace std;

using ll = long long;

int main() {
  cin.tie(0)->sync_with_stdio(0);
  cin.exceptions(cin.failbit);
}
```

Data structures (2)

BIT.h

Description: Query [l, r] sums, and point updates. kth() returns the smallest index i s.t. query(0, i) >= k

Time: $\mathcal{O}(\log n)$ for all ops.

792db7, 22 lines

```
template<typename T>
struct BIT {
  vector<T> s; int n;
  BIT(int n): s(n + 1), n(n) {}
  void update(int i, T v) {
    for (i++; i <= n ; i += i & -i) s[i] += v;
  }
  T query(int i) {
    T ans = 0;
    for (i++; i > 0; i -= i & -i) ans += s[i];
    return ans;
  }
  T query(int l, int r) { return query(r) - query(l - 1); }
  int kth(T k) { // returns n if k > sum of tree
    if (k <= 0) return -1;
    int i = 0;
    for (int pw = 1 << __lg(n); pw; pw >>= 1)
      if (i + pw <= n && s[i + pw] < k)
        k -= s[i += pw];
    return i;
  }
};
```

DSU.h

Description: Maintains union of disjoint sets

Time: $\mathcal{O}(\alpha(N))$

c22586, 14 lines

```
struct DSU {
  vector<int> s;
  DSU(int n): s(n, -1) { }
  int find(int i) { return s[i] < 0 ? i : s[i] = find(s[i]); }
  bool join(int a, int b) {
    a = find(a), b = find(b);
    if (a == b) return false;
    if (s[a] > s[b]) swap(a, b);
    s[a] += s[b], s[b] = a;
    return true;
  }
  int size(int i) { return -s[find(i)]; }
  bool same(int a, int b) { return find(a) == find(b); }
};
```

RMQ.h

Description: Constant time subarray min/max queries for a fixed array

Time: $\mathcal{O}(nlogn)$ initialization and $\mathcal{O}(1)$ queries.

536eac, 15 lines

```
template<typename T, class Compare = less<T>>
struct RMQ {
  vector<vector<T>> t;
  Compare cmp;
  RMQ(vector<T> &a) : t(__lg(a.size()) + 1, a) {
    int n = a.size(), lg = __lg(n);
    for (int k = 1, len = 1; k <= lg; k++, len <<= 1)
      for (int i = 0; i + 2*len - 1 < n; i++)
        t[k][i] = min(t[k - 1][i], t[k - 1][i + len], cmp);
  }
  T query(int a, int b) {
    int k = __lg(b - a + 1), len = 1 << k;
    return min(t[k][a], t[k][b - len + 1], cmp);
  }
};
```

Geometry (3)

Graphs (4)

Mathematics (5)

Miscellaneous (6)

NDimensionalVector.h

3c0f61, 12 lines

```
template<int D, typename T>
struct Vec : public vector<Vec<D - 1, T>> {
  static_assert(D >= 1, "Vector dimension must be greater than
       zero!");
  template<typename... Args>
  Vec(int n = 0, Args... args) : vector<Vec<D - 1, T>>(n, Vec<D
       - 1, T>(args...)) {
  }
};
template<typename T>
struct Vec<1, T> : public vector<T> {
  Vec(int n = 0, const T& val = T()) : vector<T>(n, val) {
  }
};
```

Submasks.h

35424b, 3 lines

```
for (int mask = 0; mask < (1 << n); mask++)
  for (int sub = mask; sub; sub = (sub - 1) & mask)
    // do thing
```

Strings (7)