2 買い物 2 (Shopping 2)

Author : 菅井 遼明

小課題 1

小課題 1 では，$N, M, Q \leq 2\,000$ であるから，合計金額を求めるために，その客が買った商品を列挙しても実行時間制限内に答えることができる．

時間計算量は $O(NQ)$ となる．

```cpp
#include <iostream>
#include <vector>
int main(){
  std::ios_base::sync_with_stdio(false);
  std::cin.tie(nullptr);
  int N, M, Q;
  std::cin >> N >> M >> Q;
  std::vector<int> P(N), A(N);
  for (int i = 0; i < N; i++){
    std::cin >> P[i] >> A[i];
    A[i]--;
  }
  for (int i = 0; i < Q; i++){
    int T, L, R;
    std::cin >> T >> L >> R;
    T--;
    L--;
    long long ans = 0;
    for (int j = L; j < R; j++){
      if (A[j] == T){
        ans += P[j] / 2;
      } else {
```

```
23            ans += P[j];
24          }
25        }
26        std::cout << ans << '\n';
27      }
28    }
```

小課題 2

小課題 2 では $M = 1$ である．よって，それぞれの商品の価格は必ず定価の半額となる．

商品 i $(1 \leqq i \leqq N)$ の定価を B_i とおくと，客 k $(1 \leqq k \leqq Q)$ が商品を買うのにかかった金額は $B_{L_k} + B_{L_k+1} + \cdots + B_{R_k}$ である．よって，B の累積和を前計算することで，それぞれの客について $O(1)$ 時間で答えを求めることができる．

時間計算量は $O(N + Q)$ となる．

```
1   #include <iostream>
2   #include <vector>
3   int main(){
4     std::ios_base::sync_with_stdio(false);
5     std::cin.tie(nullptr);
6     int N, M, Q;
7     std::cin >> N >> M >> Q;
8     std::vector<int> P(N), A(N);
9     for (int i = 0; i < N; i++){
10      std::cin >> P[i] >> A[i];
11      A[i]--;
12    }
13    std::vector<long long> S(N + 1);
14    S[0] = 0;
15    for (int i = 0; i < N; i++){
16      S[i + 1] = S[i] + P[i];
17    }
18    for (int i = 0; i < Q; i++){
19      int T, L, R;
20      std::cin >> T >> L >> R;
```

```
    T--;
    L--;
    std::cout << (S[R] - S[L]) / 2 << '\n';
  }
}
```

小課題 3

小課題 3 では $M \leqq 10$ である．セールのそれぞれの日について，その日の各商品の価格を求めると，小課題 2 と同様に，価格の累積和を前計算することで，その日に JOI 商店を訪れた客についての答えを求めることができる．

時間計算量は $O(NM + Q)$ となる．

```
#include <iostream>
#include <vector>
int main(){
  std::ios_base::sync_with_stdio(false);
  std::cin.tie(nullptr);
  int N, M, Q;
  std::cin >> N >> M >> Q;
  std::vector<int> P(N), A(N);
  for (int i = 0; i < N; i++){
    std::cin >> P[i] >> A[i];
    A[i]--;
  }
  std::vector<std::vector<long long>> S(M, std::vector<long long>(N + 1));
  for (int i = 0; i < M; i++){
    S[i][0] = 0;
    for (int j = 0; j < N; j++){
      if (A[j] == i){
        S[i][j + 1] = S[i][j] + P[j] / 2;
      } else {
        S[i][j + 1] = S[i][j] + P[j];
      }
    }
```

```
  }
  for (int i = 0; i < Q; i++){
    int T, L, R;
    std::cin >> T >> L >> R;
    T--;
    L--;
    std::cout << S[T][R] - S[T][L] << '\n';
  }
}
```

小課題 4

商品を買うのにかかった金額は，買った商品の定価の合計から，買った商品のうち価格が減ったものの価格の減少量の合計である．定価は客が JOI 商品を訪れる日によらないので，累積和を前計算することによりそれぞれの客について $O(1)$ 時間で求めることができる．

小課題 4 では，それぞれの商品の種類が相異なるので，それぞれの種類の商品は高々 1 個である．よって，それぞれの客が買った商品のうち価格が減ったもののは高々 1 個である．

それぞれの客 k $(1 \leqq k \leqq Q)$ について，客 k が JOI 商品に訪れた日に価格が減少した商品があるか求め，客 k がこの商品を買ったかどうか調べることにより，価格の減少量をそれぞれの客について $O(1)$ 時間で求めることができる．

時間計算量は $O(N + M + Q)$ となる．

```
#include <iostream>
#include <vector>
int main(){
  std::ios_base::sync_with_stdio(false);
  std::cin.tie(nullptr);
  int N, M, Q;
  std::cin >> N >> M >> Q;
  std::vector<int> P(N), A(N);
  for (int i = 0; i < N; i++){
    std::cin >> P[i] >> A[i];
    A[i]--;
  }
  std::vector<long long> S(N + 1);
```

```
    S[0] = 0;
    for (int i = 0; i < N; i++){
      S[i + 1] = S[i] + P[i];
    }
    std::vector<int> p(M, -1);
    for (int i = 0; i < N; i++){
      p[A[i]] = i;
    }
    for (int i = 0; i < Q; i++){
      int T, L, R;
      std::cin >> T >> L >> R;
      T--;
      L--;
      long long ans = S[R] - S[L];
      if (L <= p[T] && p[T] < R){
        ans -= P[p[T]] / 2;
      }
      std::cout << ans << '\n';
    }
  }
```

小課題 5

小課題 4 と同様に，それぞれの客について，その客が買った商品のうち価格が減ったものの価格の減少量の合計を求めることを考える．

小課題 5 では $P_i = 2$ $(1 \leqq i \leqq N)$ であるから，半額になる商品を 1 個購入するたび，価格の減少量の合計が 1 円増加する．よって，商品 $L_k, L_k + 1, \ldots, R_k$ のうち種類が T_k であるものの個数を求めればよい．

これは，以下のように解くことができる．

1. 各種類 j $(1 \leqq j \leqq M)$ について，種類 j の商品の番号をあらかじめ昇順に列挙する．
2. 種類 T_k の商品の番号を並べた列のうち，L_k 以上 R_k 以下である範囲を二分探索で求める．

時間計算量は $O(N + M + Q \log N)$ となる．

```
#include <iostream>
#include <vector>
```

```cpp
#include <algorithm>
int main(){
  std::ios_base::sync_with_stdio(false);
  std::cin.tie(nullptr);
  int N, M, Q;
  std::cin >> N >> M >> Q;
  std::vector<int> P(N), A(N);
  for (int i = 0; i < N; i++){
    std::cin >> P[i] >> A[i];
    A[i]--;
  }
  std::vector<std::vector<int>> idx(M);
  for (int i = 0; i < N; i++){
    idx[A[i]].push_back(i);
  }
  for (int i = 0; i < Q; i++){
    int T, L, R;
    std::cin >> T >> L >> R;
    T--;
    L--;
    int ans = (R - L) * 2;
    ans -= std::lower_bound(idx[T].begin(), idx[T].end(), R)
             - std::lower_bound(idx[T].begin(), idx[T].end(), L);
    std::cout << ans << '\n';
  }
}
```

小課題 6

小課題 5 の考察を発展させると，商品 $L_k, L_k + 1, \ldots, R_k$ のうち種類が T_k であるものの定価の総和を求めればよい．

小課題 5 と同様に，各種類 j $(1 \leqq j \leqq M)$ について，種類 j の商品の番号をあらかじめ昇順に列挙しておき，二分探索を行うことで，種類 T_k の商品のうち何番目から何番目まで客 k が買ったかを求めることができる．種類 T_k の商品に限定した場合でのこの範囲の定価の総和を求めればよい．各種類 j $(1 \leqq j \leqq M)$ に

ついて種類 j の商品に限定して定価の累積和を計算することで，定価の総和も高速に求めることができる．

時間計算量は $O(N + M + Q \log N)$ となる．

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
int main(){
  std::ios_base::sync_with_stdio(false);
  std::cin.tie(nullptr);
  int N, M, Q;
  std::cin >> N >> M >> Q;
  std::vector<int> P(N), A(N);
  for (int i = 0; i < N; i++){
    std::cin >> P[i] >> A[i];
    A[i]--;
  }
  std::vector<long long> S(N + 1);
  S[0] = 0;
  for (int i = 0; i < N; i++){
    S[i + 1] = S[i] + P[i];
  }
  std::vector<std::vector<int>> idx(M);
  for (int i = 0; i < N; i++){
    idx[A[i]].push_back(i);
  }
  std::vector<std::vector<long long>> sum(M);
  for (int i = 0; i < M; i++){
    int cnt = idx[i].size();
    sum[i].resize(cnt + 1);
    sum[i][0] = 0;
    for (int j = 0; j < cnt; j++){
      sum[i][j + 1] = sum[i][j] + P[idx[i][j]];
    }
  }
  for (int i = 0; i < Q; i++){
```

```
33      int T, L, R;
34      std::cin >> T >> L >> R;
35      T--;
36      L--;
37      long long ans = S[R] - S[L];
38      int l = std::lower_bound(idx[T].begin(), idx[T].end(), L) - idx[T].begin();
39      int r = std::lower_bound(idx[T].begin(), idx[T].end(), R) - idx[T].begin();
40      ans -= (sum[T][r] - sum[T][l]) / 2;
41      std::cout << ans << '\n';
42    }
43  }
```