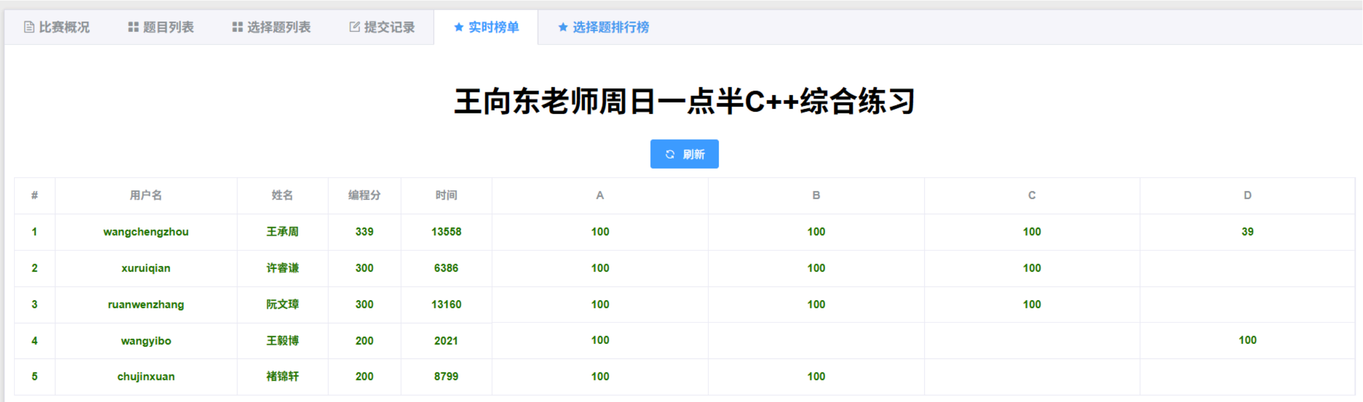
线段树

人员

王毅博、褚锦轩、王承周、许睿谦、阮文璋 到课

上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/1479



王向东老师周日一点半C++综合练习

#	用户名	姓名	编程分	时间	A	B	C	D
1	wangchengzhou	王承周	339	13558	100	100	100	39
2	xuruiqian	许睿谦	300	6386	100	100	100	
3	ruanwenzhang	阮文璋	300	13160	100	100	100	
4	wangyibo	王毅博	200	2021	100			100
5	chujinxuan	褚锦轩	200	8799	100	100		

本周作业

https://cppoj.kids123code.com/contest/1588 (课上讲了 A ~ C 题, 课后作业是 D 题)

课堂表现

今天讲了 线段树 这个知识点, 主要讲了 单点修改、区间查询 的操作

同学们课上吸收的都很好, 但 线段树 比较难的是写代码, 代码会相对复杂一些, 同学们课上要把这几道线段树的题多写几遍, 增加熟练度。

课堂内容

Strange Lunchbox (上周作业)

f[i][j]: 当获得至少 i 个章鱼丸, j 个大饼时, 最少需要多少个饭盒

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

using namespace std;

const int maxn = 300 + 5;
const int inf = 0x3f3f3f3f;
int a[maxn], b[maxn];
int f[maxn][maxn];  // f[i][j]: >=i章/>=j大 时 -> 最少多少个饭盒

int main()
{
  int n, x, y; cin >> n >> x >> y;
  for (int i = 1; i <= n; ++i) cin >> a[i] >> b[i];

  memset(f, 0x3f, sizeof(f)); f[0][0] = 0;
  for (int i = 1; i <= n; ++i) {
    for (int j = x; j >= 0; --j) {
      for (int k = y; k >= 0; --k) {
        f[j][k] = min(f[j][k], f[max(j-a[i],0)][max(k-b[i],0)]+1);
      }
    }
  }

  cout << (f[x][y]==inf ? -1 : f[x][y]) << endl;
  return 0;
}
```

I Hate It

线段树模板题, 单点修改, 区间查询最大值

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

using namespace std;

const int maxn = 2e5 + 5;
int w[maxn];
struct node {
    int l, r, maxx;
} tr[maxn*4];

void pushup(int u) { tr[u].maxx = max(tr[u*2].maxx, tr[u*2+1].maxx); }
void build(int u, int l, int r) {
    tr[u] = {l, r, 0};
    if (l == r) { tr[u].maxx = w[l]; return; }

    int mid = (l + r) / 2;
    build(u*2, l, mid), build(u*2+1, mid+1, r);
    pushup(u);
}
void modify(int u, int pos, int k) {
    if (tr[u].l == tr[u].r) { tr[u].maxx = max(tr[u].maxx, k); return; }

    int mid = (tr[u].l + tr[u].r) / 2;
    if (pos <= mid) modify(u*2, pos, k);
    else modify(u*2+1, pos, k);
    pushup(u);
}
int query(int u, int l, int r) {
    if (tr[u].l>=l && tr[u].r<=r) return tr[u].maxx;

    int mid = (tr[u].l + tr[u].r) / 2;
    int res = 0;
    if (l <= mid) res = query(u*2, l, r);
    if (r > mid) res = max(res, query(u*2+1, l, r));
    return res;
}

int main()
{
    int n, m; cin >> n >> m;
    for (int i = 1; i <= n; ++i) cin >> w[i];
    build(1, 1, n);

    while (m -- ) {
        char op[2]; cin >> op;
        if (op[0] == 'Q') {
            int l, r; cin >> l >> r;
            cout << query(1, l, r) << endl;
        } else {
            int pos, k; cin >> pos >> k;
            modify(1, pos, k);
        }
    }
    return 0;
}
```

统计和

线段树模板题, 单点修改, 区间查询区间和

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

using namespace std;

typedef long long LL;
const int maxn = 1e5 + 5;
struct node {
  int l, r;
  LL sum;
} tr[maxn*4];

void pushup(int u) { tr[u].sum = tr[u*2].sum + tr[u*2+1].sum; }

void build(int u, int l, int r) {
  tr[u] = {l, r, 0};
  if (l == r) return;
  int mid = (l + r) / 2;
  build(u*2, l, mid), build(u*2+1, mid+1, r);
}

void modify(int u, int pos, int k) {
  if (tr[u].l==pos && tr[u].r==pos) {
    tr[u].sum += k; return;
  }

  int mid = (tr[u].l + tr[u].r) / 2;
  if (pos <= mid) modify(u*2, pos, k);
  else modify(u*2+1, pos, k);
  pushup(u);
}

LL query(int u, int l, int r) {
  if (tr[u].l>=l && tr[u].r<=r) return tr[u].sum;

  LL res = 0;
  int mid = (tr[u].l + tr[u].r) / 2;
  if (l <= mid) res += query(u*2, l, r);
  if (r > mid) res += query(u*2+1, l, r);
  return res;
}

int main()
{
  int n, m; cin >> n >> m;
  build(1, 1, n);
  while (m -- ) {
    char op[2]; cin >> op;
    if (op[0] == 'x') {
      int a, b; cin >> a >> b; modify(1, a, b);
    } else {
      int l, r; cin >> l >> r;
//      cout << "------------------- ";
      cout << query(1, l, r) << "\n";
    }
  }
  return 0;
}
```

[JSOI2008] 最大数

本质上也是单点修改, 区间查询最大值

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

using namespace std;

typedef long long LL;
const int maxn = 2e5 + 5;
const LL inf = 0x3f3f3f3f3f3f3f3f;
struct node {
  int l, r;
  LL maxx;
} tr[maxn*4];
```

```
void pushup(int u) {
  tr[u].maxx = max(tr[u*2].maxx, tr[u*2+1].maxx);
}

void build(int u, int l, int r) {
  tr[u] = {l, r, -inf};
  if (l == r) return;
  int mid = (l + r) / 2;
  build(u*2, l, mid), build(u*2+1, mid+1, r);
}

void modify(int u, int pos, LL k) {
  if (tr[u].l==pos && tr[u].r==pos) {
    tr[u].maxx = k; return;
  }

  int mid = (tr[u].l+tr[u].r) / 2;
  if (pos <= mid) modify(u*2, pos, k);
  else modify(u*2+1, pos, k);
  pushup(u);
}

LL query(int u, int l, int r) {
  if (tr[u].l>=l && tr[u].r<=r) return tr[u].maxx;

  int mid = (tr[u].l+tr[u].r) / 2;
  LL res = -inf;
  if (l <= mid) res = query(u*2, l, r);
  if (r > mid) res = max(res, query(u*2+1, l, r));
  return res;
}

int main()
{
  int n; LL mod; cin >> n >> mod;
  build(1, 1, n);
  LL t = 0;
  for (int i = 1, id = 0; i <= n; ++i) {
    char op[2]; cin >> op;
    if (op[0] == 'A') {
      LL n; cin >> n;
      n = (n + t) % mod;
      ++id;
      modify(1, id, n);
    } else {
      int k; cin >> k;
      t = query(1, id-k+1, id);

      cout << t << endl;
    }
  }
```

```
    return 0;
}
```