Team notebook

Nicolas Alba

June 15, 2024

Contents

1 Karatsuba

```
#define ll long long
const int MOD = 1e9+7;
#define poly vector<ll>
#define fore(i,a,b) for(int i=a,ThxDem=b;i<ThxDem;++i)
typedef int tp;

ll sum(ll x, ll y) {
   ll ans = (x + y) % MOD;
   if (ans < 0) ans += MOD;
   return ans;
}

ll mult(ll x, ll y) {
   ll ans = (x % MOD) * (y % MOD);
   ans %= MOD;
   if (ans < 0) ans += MOD;
   return ans;
}

#define add(n,s,d,k) fore(i,0,n)(d)[i]=sum((d)[i], mult((s)[i],k))
tp* ini(int n){tp *r=new tp[n];fill(r,r+n,0);return r;}
void karatsura(int n, tp* p, tp* q, tp* r){
      if(n<=0)return;
      if(n<35)fore(i,0,n)fore(j,0,n)r[i+j]=sum(r[i+j], mult(p[i],q[j]));
      else {
            int nac=n/2,nbd=n-n/2;
            tp *a=p,*b=p+nac,*c=q,*d=q+nac;
            tp
                *ab=ini(nbd+1),*cd=ini(nbd+1),*ac=ini(nac*2),*bd=ini(nbd*2);
            add(nac,a,ab,1);add(nbd,b,ab,1);
            add(nac,c,cd,1);add(nbd,d,cd,1);
            karatsura(nac,a,c,ac);karatsura(nbd,b,d,bd);
            add(nac*2,ac,r+nac,-1);add(nbd*2,bd,r+nac,-1);
            add(nac*2,ac,r,1);add(nbd*2,bd,r+nac*2,1);
            karatsura(nbd+1,ab,cd,r+nac);
            free(ab);free(cd);free(ac);free(bd);
      }
}
vector<tp> multiply(vector<tp> p0, vector<tp> p1){
      int n=max(p0.size(),p1.size());
      tp *p=ini(n),*q=ini(n),*r=ini(2*n);
      fore(i,0,p0.size())p[i]=p0[i];
      fore(i,0,p1.size())q[i]=p1[i];
      karatsura(n,p,q,r);
      vector<tp> rr(r,r+p0.size()+p1.size()-1);
      free(p);free(q);free(r);
      return rr;
}
```

2 **general**$_{i}terative_{s}egtree$

```
// >>>>>>>>> Implement
struct Node { ll x = 0; };
```

```
Node e() { return Node(); }

Node op(Node &a, Node &b) {
    Node c;
    c.x = a.x + b.x;
    return c;
}
// <<<<<<<

struct segtree {
    vector<Node> t;
    ll n;

    void init(int n) {
        t.assign(n * 2, e());
        this->n = n;
    }

    void init(vector<Node>& s) {
        n = s.size();
        t.assign(n * 2, e());
        for (int i = 0; i < n; i++) {
            t[i+n] = s[i];
        }
        build();
    }

    void build() { // build the tree
        for (int i = n - 1; i > 0; --i) t[i] = op(t[i<<1], t[i<<1|1]);
    }

    // set value at position p
    void update(int p, const Node& value) {
        for (t[p += n] = value; p >>= 1; ) t[p] = op(t[p<<1], t[p<<1|1]);
    }

    // sum on interval [l, r]
    Node query(int l, int r) {
        r++; // make this inclusive
        Node resl=e(), resr=e(); // null element
        for (l += n, r += n; l < r; l >>= 1, r >>= 1) {
            if (l&1) resl = op(resl, t[l++]);
            if (r&1) resr = op(t[--r], resr);
        }
        return op(resl, resr);
    }

    Node get(int i) {
        return query(i, i);
    }
};
```