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

using namespace std;

using ll = long long;
using vi = vector<int>;
using vll = vector<ll>;
using vvi = vector<vector<int>>;
using vvl = vector<vector<ll>>;

const int MAX_STRING_SIZE = 1000000;

ll base = 1007;

vector<ll> MODS({2147483647, 6700417, 524287, 1000000007}

);

vector<vector<ll>> invs(MODS.size()), pows(MODS.size());

ll powm(ll e, ll x, ll m) {
  if (x == 0) return 1;
  if (x == 1) return e;
  if (x % 2 == 0) return powm((e * e) % m, x / 2, m);
  return (e * powm(e, x - 1, m)) % m;
}

void RHinit() {
  int i = 0;
  for (auto &p : MODS) {
    invs[i].resize(MAX_STRING_SIZE + 1);
    invs[i][0] = 1;
    ll inv = powm(base, p - 2, p);
    for (int j = 1; j <= MAX_STRING_SIZE; j++) {
      invs[i][j] = (invs[i][j - 1] * inv) % p;
    }
    pows[i].resize(MAX_STRING_SIZE + 1);
    pows[i][0] = 1;
    for (int j = 1; j <= MAX_STRING_SIZE; j++) {
      pows[i][j] = (pows[i][j - 1] * base) % p;
    }
    i++;
  }
}

struct RH {
  int n;
  string s;
  vector<vector<ll>> hs;

  RH(string &s) : s(s) {
    if (pows[0].empty()) {
      RHinit();
    }
    n = s.size();
    hs.resize(MODS.size());
    for (int i = 0; i < MODS.size(); i++) {
      hs[i].resize(n + 1);
      hs[i][0] = 0;
      for (int j = 0; j < n; j++) {
        hs[i][j + 1] = (hs[i][j] + pows[i][j] * s[j]) % MODS[i];
      }
    }
  }

  vector<ll> substr(int l, int r) {
    vector<ll> ret(MODS.size());
    for (int i = 0; i < MODS.size(); i++) {
      ret[i] = (hs[i][r] - hs[i][l] + MODS[i]) % MODS[i];
      ret[i] *= invs[i][l];
      ret[i] %= MODS[i];
    }
    return ret;
  }
};

struct N {
  vector<ll> v;
};

bool operator<(const N &l, const N &r) {
  if (l.v[0] == r.v[0]) {
    if (l.v[1] == r.v[1]) {
      if (l.v[2] == r.v[2]) {
        return l.v[3] < r.v[3];
      }
      return l.v[2] < r.v[2];
    }
    return l.v[1] < r.v[1];
  }
  return l.v[0] < r.v[0];
}

int main() {
  int n, m;
  cin >> n >> m;
  string s;
  cin >> s;
  RH h(s);
  int l = 0, r = 1;
  set<N> ss;
  for (int i = 0; i < m; i++) {
    string t;
    cin >> t;
    if (t == "L++") {
      l++;
      r;
    } else if (t == "L--") {
      l--;
      r;
    } else if (t == "R++") {
      l;
      r++;
    } else if (t == "R--") {
      l;
      r--;
    }
    ss.insert(N{h.substr(l, r)});
  }
  cout << ss.size() << endl;
  return 0;
}

```