Crack – a – Hack

Animal transport

Course: ALGORITHMIC PROBLEM SOLVING

Course Code: 17ECSE309

By: Anoop kulkarni

01FE16BCS404

Problem:

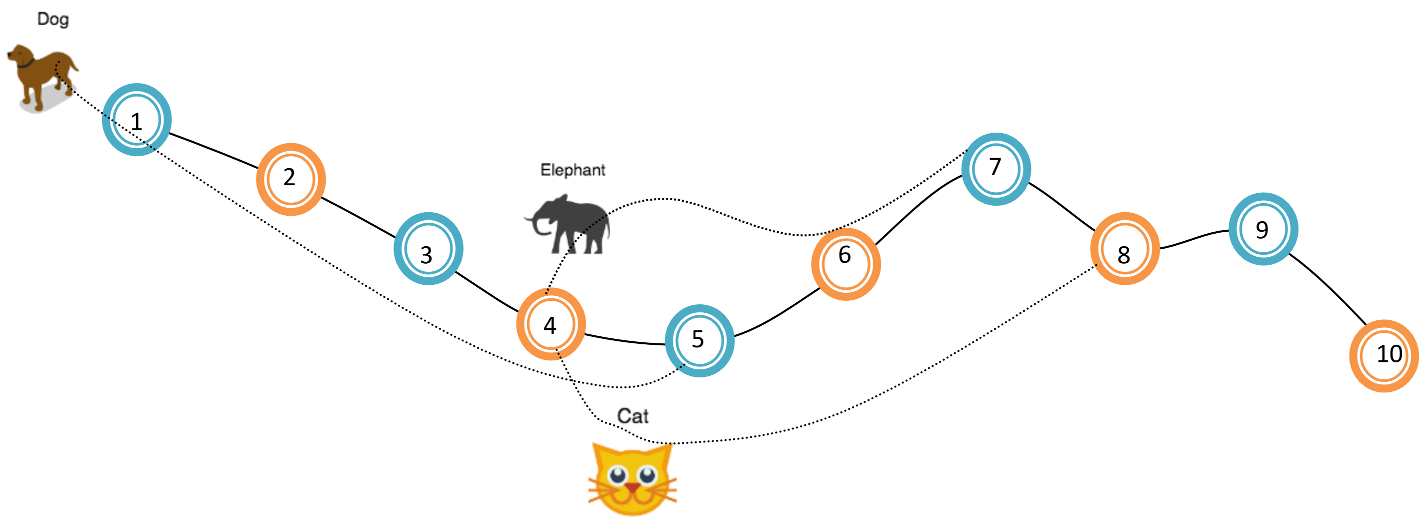
Capeta is working part-time for an animal shipping company. He needs to pick up animals from various zoos and drop them to other zoos. The company ships four kinds of animals: elephants, dogs, cats, and mice.

There are zoos, numbered to. Also, there are animals. For each animal, Capeta knows its type (E for elephant, D for dog, C for cat and M for mouse), source zoo where Capeta has to pick it up from, and destination zoo where Capeta needs to deliver it to. Capeta is given a truck with a huge capacity where animals can easily fit. He is also given additional instructions.

1. He must visit the zoos in increasing order. He also cannot skip zoos.
2. Dogs are scared of elephants, so he is not allowed to bring them together at the same time.
3. Cats are scared of dogs, so he is not allowed to bring them together at the same time.
4. Mice are scared of cats, so he is not allowed to bring them together at the same time.
5. Elephants are scared of mice, so he is not allowed to bring them together at the same time.

Example:

If there are 10 zoos and 3 animals to be transported.



Capeta can transport one animal by traveling up to zoo number . Just drop the dog there. Next, in order to transport animals (elephant and cat), Capeta has to go up to zoo number .

Hackerrank Question Link:

[world-codesprint-12 : Animal-transport]

Solution:

The solution is written in c++:

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

#define forn(i,n) for (int i = 0; i < int(n); ++i)
#define pb push_back
#define mp make_pair
#define sz(a) int(a.size())
#define all(a) a.begin(),a.end()

typedef pair<int,int> pt;
#define ft first
#define sc second

typedef long long li;
typedef long double ld;

using namespace std;

bool solve(int);

int main() {
#ifdef SU1
    freopen("input.txt", "r", stdin);
//  freopen("output.txt", "w", stdout);
#endif
    int T;
    cin >> T;

    int test = 0;
    while (solve(test++));

    return 0;
}

const int N = 100005;
const int S = 150;
const int B = N / S;
const string anims = "EDCM";
int m, n;
int p[N], l[N], r[N], t[N];

bool cmp(int i, int j) {
    return l[i] < l[j];
}

int a[2][N], b[2][B];
```

```
pt upd[2][B];

int xs[N], szxs;

void push(int t, int v) {
    if (upd[t][v] == mp(0, 0))
        return;
    int st = v * S;
    forn(i, S)
        a[t][st + i] = max(a[t][st + i] + upd[t][v].sc, upd[t][v].ft);
    upd[t][v] = mp(0, 0);
}

void recalc(int t, int v) {
    if (b[t][v] != -1)
        return;
    int st = v * S;
    forn(i, S)
        b[t][v] = max(b[t][v], a[t][st + i]);
}

int getp(int t, int v) {
    push(t, v / S);
    return a[t][v];
}

void inclr(int t, int l, int r, int delta) {
    while (l < r) {
        if (l % S == 0 && l + S - 1 < r) {
            recalc(t, l / S);
            upd[t][l / S].sc += delta;
            if (upd[t][l / S].ft > 0)
                upd[t][l / S].ft += delta;
            l += S;
        } else {
            push(t, l / S);
            b[t][l / S] = -1;
            a[t][l] += delta;
            l++;
        }
    }
}

void updlr(int t, int l, int r, int val) {
    while (l < r) {
        if (l % S == 0 && l + S - 1 < r) {
            recalc(t, l / S);
            upd[t][l / S].ft = max(upd[t][l / S].ft, val);
            l += S;
        } else {
            push(t, l / S);
            b[t][l / S] = -1;
            a[t][l] = max(a[t][l], val);
            l++;
        }
    }
}

int res[N];

bool solve(int) {
    if (scanf("%d %d", &m, &n) != 2)
        return false;

    szxs = 0;
    forn(i, n) {
        char buf[3];
        scanf("%s", buf);
        t[i] = anims.find(buf[0]) & 1;
        p[i] = i;
    }
    forn(i, n) {
        scanf("%d", &l[i]);
        xs[szxs++] = l[i];
    }
    forn(i, n) {
        scanf("%d", &r[i]);
        xs[szxs++] = r[i];
    }

    memset(a, 0, sizeof(a));
    memset(upd, 0, sizeof(upd));
    memset(b, -1, sizeof(b));

    sort(xs, xs + szxs);
    szxs = unique(xs, xs + szxs) - xs;
    forn(i, n) {
        l[i] = lower_bound(xs, xs + szxs, l[i]) - xs;
        r[i] = lower_bound(xs, xs + szxs, r[i]) - xs;
    }

    sort(p, p + n, cmp);

    forn(ti, n) {
        int i = p[ti];
        if (l[i] >= r[i]) continue;
        int val = getp(!t[i], l[i]);
        inclr(t[i], r[i], szxs, +1);
        updlr(t[i], r[i], szxs, val + 1);
    }
    forn(i, n)
        res[i + 1] = szxs;
    for (int i = szxs - 1; i >= 0; --i)
        forn(t, 2)
            res[getp(t, i)] = i;

    xs[szxs] = -1;
    for (int i = n; i > 0; --i)
        if (i + 1 < n)
            res[i] = min(res[i], res[i + 1]);

    forn(i, n) {
        if (i)
            putchar(' ');
        printf("%d", xs[res[i + 1]]);
    }
    puts("");
    return true;
}
```

➢ **Time Complexity:**

It takes a time complexity of O(n log n).

➢ **References:**

1. https://www.hackerrank.com/contests/world-codesprint-12/challenges/animal-transport/
2. https://discuss.codechef.com/questions/119853/hackerrank-worldcodesprint-problem-5