A

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

using namespace std;

int main() {

    freopen("input.txt", "r", stdin);

    freopen("output.txt", "w", stdout);

    int n, k, t, max;

    scanf("%i%i", &n, &k);

    vector <int> a, index;

    a.reserve(n);

    index.reserve(n);

    a.push\_back(0);

    index.push\_back(-1);

    for (int i = 1; i < n; i++) {

        max = i - 1;

        if (i < n - 1) {

            scanf("%i", &t);

        }

        else {

            t = 0;

        }

        for (int j = i - 2; j >= 0 && j >= i - k; j--) {

            if (a.at(j) > a.at(max)) {

                max = j;

            }

        }

        a.push\_back(t + a.at(max));

        index.push\_back(max);

    }

    printf("%i\n", a.back());

    a.clear();

    t = n - 1;

    while(t >= 0) {

        a.push\_back(t);

        t = index.at(t);

    }

    printf("%i\n", a.size() - 1);

    for (int i = a.size() - 1; i >= 0; i--) {

        printf("%i ", a.at(i) + 1);

    }

    return 0;

}

B

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

using namespace std;

int indexOf(long long a, vector <long long> dp) {

    int l = -1, r = dp.size(), m;

    while (l < r - 1) {

        m = (l + r) / 2;

        if (dp.at(m) < a) {

            l = m;

        }

        else {

            r = m;

        }

    }

    return r;

}

int main() {

    int n;

    long long t;

    scanf("%i", &n);

    vector <long long> a, index, prev, dp;

    index.reserve(n + 1);

    prev.reserve(n);

    dp.reserve(n + 1);

    a.reserve(n);

    dp.push\_back(LLONG\_MIN);

    for (int i = 0; i <= n; i++) {

        index.push\_back(-1);

    }

    for (int i = 0; i < n; ++i) {

        scanf("%lld", &t);

        a.push\_back(t);

        prev.push\_back(-1);

        t = indexOf(t, dp);

        if (dp.at(t - 1) < a.back() && (t >= dp.size() || dp.at(t) > a.back())) {

            if (t < dp.size()) {

                dp.at(t) = a.back();

            }

            else {

                dp.push\_back(a.back());

            }

            index.at(t) = i;

            prev.at(i) = index.at(t - 1);

        }

    }

    printf("%i\n", dp.size() - 1);

    t = index.at(dp.size() - 1);

    index.clear();

    while (t > -1) {

        index.push\_back(a.at(t));

        t = prev.at(t);

    }

    for (int i = index.size() - 1; i >= 0; --i) {

        printf("%lld ", index.at(i));

    }

    return 0;

}

C

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

using namespace std;

int main() {

    int n;

    long long sum = 0;

    scanf("%i", &n);

    vector <long long> last;

    vector <long long> current;

    last.reserve(10);

    current.reserve(10);

    for (int i = 0; i < 10; i++) {

        last.push\_back(1);

        current.push\_back(1);

    }

    last.at(0) = last.at(8) = 0;

    for (int i = 1; i < n; i++) {

        current.at(1) = (last.at(6) + last.at(8)) % 1000000000;

        current.at(2) = (last.at(7) + last.at(9)) % 1000000000;

        current.at(3) = (last.at(4) + last.at(8)) % 1000000000;

        current.at(4) = (last.at(3) + last.at(9) + last.at(0)) % 1000000000;

        current.at(5) = 0;

        current.at(6) = (last.at(1) + last.at(7) + last.at(0)) % 1000000000;

        current.at(7) = (last.at(2) + last.at(6)) % 1000000000;

        current.at(8) = (last.at(1) + last.at(3)) % 1000000000;

        current.at(9) = (last.at(2) + last.at(4)) % 1000000000;

        current.at(0) = (last.at(4) + last.at(6)) % 1000000000;

        last = current;

    }

    for (int i = 0; i < 10; ++i) {

        sum += last.at(i);

        sum %= 1000000000;

    }

    printf("%lld", sum);

    return 0;

}

D

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int main() {

    freopen("input.txt", "r", stdin);

    freopen("output.txt", "w", stdout);

    string a, b;

    cin >> a >> b;

    int \*\* dp = new int \*[a.length() + 1];

    for (int i = 0; i <= a.length(); ++i) {

        dp[i] = new int[b.length() + 1];

        dp[i][0] = i;

    }

    for (int j = 0; j <= b.length(); ++j) {

        dp[0][j] = j;

    }

    for (int i = 1; i <= a.length(); ++i) {

        for (int j = 1; j <= b.length(); ++j) {

            if (a[i - 1] == b[j - 1]) {

                dp[i][j] = dp[i - 1][j - 1];

            }

            else {

                dp[i][j] = min(dp[i][j - 1], dp[i - 1][j]);

                dp[i][j] = min(dp[i][j], dp[i - 1][j - 1]);

                dp[i][j]++;

            }

        }

    }

    printf("%i", dp[a.length()][b.length()]);

    return 0;

}

E

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int main() {

    int n, k;

    int min\_value = 300000, min\_index = 0;

    vector <int> a, days;

    scanf("%i", &n);

    if (n == 0) {

        printf("0\n0 0\n");

        return 0;

    }

    a.reserve(n);

    int \*\* b = new int \*[n];

    for (int i = 0; i < n; i++) {

        scanf("%i", &k);

        a.push\_back(k);

        b[i] = new int[n + 1];

        for (int j = 0; j <= n; ++j) {

            b[i][j] = 300000;

        }

    }

    if (a[0] > 100) {

        b[0][1] = a.at(0);

    }

    else {

        b[0][0] = a.at(0);

    }

    for (int i = 1; i < n; ++i) {

        for (int j = 0; j <= n; ++j) {

            if (a.at(i) <= 100) {

                b[i][j] = min(j < n ? b[i - 1][j + 1] : 300000, b[i - 1][j] + a.at(i));

            }

            else {

                b[i][j] = min(j < n ? b[i - 1][j + 1] : 300000, j > 0 ? b[i - 1][j - 1] + a.at(i) : 300000);

            }

        }

    }

    for (int j = 0; j <= n; ++j) {

        if (b[n - 1][j] <= min\_value) {

            min\_value = b[n - 1][j];

            min\_index = j;

        }

    }

    k = min\_index;

    for (int i = n - 1; i > 0; --i) {

        if (k > 0 && a.at(i) > 100 && b[i][k] == b[i - 1][k - 1] + a.at(i)) {

            --k;

        }

        else {

            if (!(a.at(i) <= 100 && b[i][k] == b[i - 1][k] + a.at(i))) {

                days.push\_back(i + 1);

                ++k;

            }

        }

    }

    printf("%i\n%i %i\n", min\_value, min\_index, days.size());

    for (int i = days.size() - 1; i >= 0; --i) {

        printf("%i\n", days.at(i));

    }

    return 0;

}

F

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int main() {

    string a;

    cin >> a;

    int n = a.size();

    int \*\* dp = new int \*[n];

    for (int i = 0; i < n; i++) {

        dp[i] = new int[n];

        for (int j = 0; j < i; ++j) {

            dp[i][j] = 0;

        }

        dp[i][i] = 1;

    }

    for (int i = 1; i < n; i++) {

        for (int j = 0; j < n - i; ++j) {

            dp[j][j + i] = min(dp[j][j + i - 1] + 1, dp[j + 1][j + i] + 1);

            dp[j][j + i] = min(dp[j][j + i], dp[j + 1][j + i - 1] + 2);

            for (int k = j; k < j + i; ++k) {

                dp[j][j + i] = min(dp[j][j + i], dp[j][k] + dp[k + 1][j + i]);

            }

            if (a[j] == '(' && a[j + i] == ')' ||

                a[j] == '[' && a[j + i] == ']' ||

                a[j] == '{' && a[j + i] == '}') {

                dp[j][j + i] = min(dp[j][j + i], dp[j + 1][j + i - 1]);

            }

        }

    }

    printf("%i", n - dp[0][n - 1]);

    return 0;

}

G

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int main() {

    string a;

    cin >> a;

    int n = a.size();

    int \*\* dp = new int \*[n];

    string \*\* s = new string \*[n];

    for (int i = 0; i < n; i++) {

        dp[i] = new int[n];

        s[i] = new string[n];

        for (int j = 0; j < i; ++j) {

            dp[i][j] = 0;

            s[i][j] = "";

        }

        dp[i][i] = 1;

        s[i][i] =  "";

    }

    for (int i = 1; i < n; i++) {

        for (int j = 0; j < n - i; ++j) {

            if (dp[j][j + i - 1] < dp[j + 1][j + i]) {

                s[j][j + i] = s[j][j + i - 1];

                dp[j][j + i] = dp[j][j + i - 1] + 1;

            }

            else {

                s[j][j + i] = s[j + 1][j + i];

                dp[j][j + i] = dp[j + 1][j + i] + 1;

            }

            if (dp[j][j + i] > dp[j + 1][j + i - 1] + 2) {

                dp[j][j + i] = dp[j + 1][j + i - 1] + 2;

                s[j][j + i] = s[j + 1][j + i - 1];

            }

            for (int k = j; k < j + i; ++k) {

                if (dp[j][j + i] > dp[j][k] + dp[k + 1][j + i]) {

                    dp[j][j + i] = dp[j][k] + dp[k + 1][j + i];

                    s[j][j + i] = s[j][k] + s[k + 1][j + i];

                }

            }

            if ((a[j] == '(' && a[j + i] == ')' ||

                a[j] == '[' && a[j + i] == ']' ||

                a[j] == '{' && a[j + i] == '}') &&

                dp[j][j + i] > dp[j + 1][j + i - 1]) {

                dp[j][j + i] = dp[j + 1][j + i - 1];

                if (a[j] == '(') {

                    s[j][j + i] = "(" + s[j + 1][j + i - 1] + ")";

                }

                if (a[j] == '[') {

                    s[j][j + i] = "[" + s[j + 1][j + i - 1] + "]";

                }

                if (a[j] == '{') {

                    s[j][j + i] = "{" + s[j + 1][j + i - 1] + "}";

                }

            }

        }

    }

    cout << s[0][n - 1];

    return 0;

}

H

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int l(int n, vector<int> \* children, int \* prel) {

    if (prel[n] != -1) {

        return prel[n];

    }

    int ch = 0, grch = 0, k;

    for (int i = 0; i < children[n].size(); ++i) {

        k = children[n].at(i);

        ch += prel[k] == - 1 ? l(k, children, prel) : prel[k];

        for (int j = 0; j < children[k].size(); ++j) {

            grch += prel[children[k].at(j)] == -1 ? l(children[k].at(j), children, prel) : prel[children[k].at(j)];

        }

    }

    prel[n] = max(ch, grch + 1);

    return prel[n];

}

int main() {

    int n, k;

    scanf("%i", &n);

    vector<int> \* children = new vector<int>[n + 1];

    int \* prel = new int[n + 1];

    for (int i = 0; i < n; ++i) {

        scanf("%i", &k);

        children[k].push\_back(i + 1);

        prel[i + 1] = -1;

    }

    printf("%i", l(children[0].front(), children, prel));

    return 0;

}

J

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

long long shortestLength(long long \*\* dp, long long \*\* d, int n, int i, int mask) {

    if (dp[i][mask] != 130000000) {

        return dp[i][mask];

    }

    for (int j = 0; j < n; j++) {

        if (mask & (1 << j)) {

            dp[i][mask] = min(dp[i][mask], shortestLength(dp, d, n, j, mask - (1 << j)) + d[i][j]);

        }

    }

    return dp[i][mask];

}

int main() {

    int n, k, minind = 0;

    long long min = LLONG\_MAX;

    scanf("%i", &n);

    int mask = (1 << n) - 1;

    vector<int> answer;

    answer.reserve(n);

    long long \*\* d = new long long \*[n];

    long long \*\* dp = new long long \*[n];

    for (int i = 0; i < n; i++) {

        d[i] = new long long[n];

        dp[i] = new long long[mask + 1];

        for (int j = 0; j < n; j++) {

            scanf("%i", &k);

            d[i][j] = k;

        }

        dp[i][0] = 0;

        for (int j = 1; j <= mask; j++) {

            dp[i][j] = 130000000;

        }

    }

    for (int i = 0; i < n; i++) {

        shortestLength(dp, d, n, i, mask - (1 << i));

        if (dp[i][mask - (1 << i)] <= min) {

            min = dp[i][mask - (1 << i)];

            minind = i;

        }

    }

    printf("%lld\n%i ", min, minind + 1);

    mask -= (1 << minind);

    for (int j = 0; j < n && mask; j++) {

            if (mask & (1 << j) &&

                dp[minind][mask] == dp[j][mask - (1 << j)] + d[minind][j]) {

                printf("%i ", j + 1);

                mask -= (1 << j);

                minind = j;

                j = -1;

            }

        }

    return 0;

}

K

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int main() {

    freopen("nice.in", "r", stdin);

    freopen("nice.out", "w", stdout);

    int n, m, black, white;

    int nice;

    scanf("%i%i", &n, &m);

    vector <int> current, last;

    if (n > m) {

        swap(n, m);

    }

    for (int i = 0; i < (1 << n); ++i) {

        last.push\_back(1);

        current.push\_back(0);

    }

    for (int t = 1; t < m; ++t) {

        for (int i = 0; i < (1 << n); ++i) {

            current.at(i) = 0;

            for (int j = 0; j < (1 << n); ++j) {

                black = i & j;

                white = (~(i | j));

                nice = 1;

                for (int k = 0; k <= n - 2; ++k) {

                    if ((black & (3 << k)) == (3 << k) || (white & (3 << k)) == (3 << k)) {

                        nice = 0;

                        break;

                    }

                }

                current.at(i) += last.at(j) \* nice;

            }

        }

        last = current;

    }

    long long answer = 0;

    for (int i = 0; i < (1 << n); ++i) {

        answer += last.at(i);

    }

    printf("%lld", answer);

    return 0;

}

I

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include <iostream>

#include <math.h>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

int main() {

    freopen("skyscraper.in", "r", stdin);

    freopen("skyscraper.out", "w", stdout);

    int n, w, s;

    scanf("%i%i", &n, &w);

    vector <int> answer;

    int \* a = new int[n];

    int \* dp = new int [n + 1];

    vector <int> sum;

    dp[0] = 0;

    for (int i = 0; i < n; ++i) {

        scanf("%i", &a[i]);

    }

    for (int j = 0; j < 1 << n; ++j) {

        s = 0;

        for (int i = 0; i < n; ++i) {

            if (j & (1 << i)) {

                s += a[i];

            }

        }

        sum.push\_back(s);

    }

    for (int i = 1; i <= n; ++i) {

        dp[i] = 0;

        for (int j = 1; j < 1 << n; ++j) {

            if (sum[j] > sum[dp[i]] && sum[j] - sum[dp[i - 1]] <= w && (j | dp[i - 1]) == j) {

                dp[i] = j;

            }

        }

        if (dp[i] == (1 << n) - 1) {

            printf("%i", i);

            while (dp[i]) {

                w = dp[i--] ^ dp[i - 1];

                for (int k = 0; k < n; ++k) {

                    if (w & (1 << k)) {

                        answer.push\_back(k + 1);

                    }

                }

                printf("\n%i ", answer.size());

                for (int k = 0; k < answer.size(); ++k) {

                    printf("%i ", answer.at(k));

                }

                answer.clear();

            }

            break;

        }

    }

    return 0;

}