算法设计与应用基础：Homework No.3

16337341 朱志儒

1. Two Buttons

Vasya has found a strange device. On the front panel of a device there are: a red button, a blue button and adisplay showing some positive integer. After clicking the red button, device multiplies the displayed number by two. After clicking the blue button, device subtracts one from the number on the display. If at some point the number stops being positive, the device breaks down. The display can show arbitrarily large numbers. Initially, the display shows number n. Bob wants to get number m on the display. What minimum number of clicks he has to make in order to
achieve this result?

Input

The first and the only line of the input contains two distinct integers n and m ($1 \le n, m \le 104$), separated by a space .

Output

Print a single number — the minimum number of times one needs to push the button required to get the number m out of number n.

Examples

input
4 6
output
2

input
10 1
output
9

Note

In the first example you need to push the blue button once, and then push the red button once.

In the second example, doubling the number is unnecessary, so we need to push the blue button nine
times.

解题思路：

从 m 推向 n，如果 m < n 则直接输出 n – m，否则开始计数。首先判断 m 是

否为奇数，若为奇数则 m++，若为偶数则判断 m，n 的大小，若 m > n 则 m /= 2，

若 m < n 则 m++，每对 m 进行一次操作就将 count++，直至 m == n，输出 count。

代码如下：

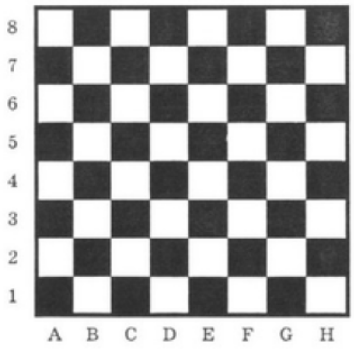
```
#include <iostream>
using namespace std;
int main() {
    int n, m;
    cin >> n >> m;
    if (m < n) {
        cout << n - m;
        return 0;}
    int count = 0;
    while (m != n) {
        if (m % 2 == 1) {
            m++;
            count++;}
        else {
            if (m > n) {
                m /= 2;
                count++;}
            if (m < n) {
                m++;
                count++;}}}
    cout << count;
    return 0;}
```

结果：

38908919	2018-06-03 07:10:28	zhuzhr3	520B - Two Buttons	GNU C++17	Accepted	15 ms	3300 KB

2. Shortest path of the king

The king is left alone on the chessboard. In spite of this loneliness, he doesn't lose heart, because he has business of national importance. For example, he has to pay an official visit to square t. As the king is not in habit of wasting his time, he wants to get from his current position s to square t in the least number of moves. Help him to do this.



In one move the king can get to the square that has a common side or a common vertex with the square the king is currently in (generally there are 8 different squares he can move to).

Input

The first line contains the chessboard coordinates of square s, the second line — of square t.

Chessboard coordinates consist of two characters, the first one is a lowercase Latin letter (from a to h), the second one is a digit from 1to 8.

Output

In the first line print n — minimum number of the king's moves. Then in n lines print the moves themselves. Each move is described with one of the 8: L, R, U, D, LU, LD, RU or RD.

L, R, U, D stand respectively for moves left, right, up and down (according to the picture), and 2-letter combinations stand for diagonal moves. If the answer is not unique, print any of them.

Examples

input
a8
h1
output
7
RD
RD
RD
RD
RD
RD
RD

解题思路：

计算 t 到 s 的横纵坐标的差值，两个中较大的那个是总移动步数，较小的那个是走对角线的总步数，再根据 s 与 t 的方位判断棋子的移动方式。

代码如下：

```
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
    char s[2], t[2];
    cin >> s >> t;
    int x = t[0] - s[0], y = t[1] - s[1];
    int diagonal = min(abs(x), abs(y));
    int step = max(abs(x), abs(y));
    int side = step - diagonal;
    cout << step << endl;
    if (diagonal != 0 && x > 0 && y < 0) {
        for (int i = 0; i < diagonal; ++i) {
            cout << "RD" << endl;}}
    else if (diagonal != 0 && x > 0 && y > 0) {
        for (int i = 0; i < diagonal; ++i) {
            cout << "RU" << endl;}}
    else if (diagonal != 0 && x < 0 && y < 0) {
        for (int i = 0; i < diagonal; ++i) {
            cout << "LD" << endl;}}
    else if (diagonal != 0 && x < 0 && y > 0) {
        for (int i = 0; i < diagonal; ++i) {
            cout << "LU" << endl;}}
    if (side != 0) {
        if (abs(x) > abs(y)) {
            if (x > 0) {
                for (int i = 0; i < side; ++i) {
                    cout << 'R' << endl;}}
            else {
                if (x < 0) {
                    for (int i = 0; i < side; ++i) {
                        cout << 'L' << endl;}}}}
        else if (abs(x) < abs(y)) {
            if (y > 0) {
                for (int i = 0; i < side; ++i) {
                    cout << 'U' << endl;}}
            else if (y < 0) {
                for (int i = 0; i < side; ++i) {
                    cout << 'D' << endl;}}}}
    return 0;}
```

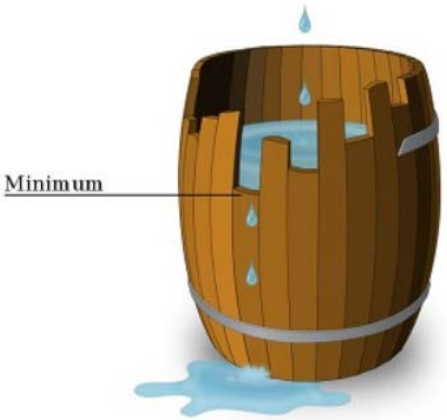
结果：

38771301	2018-05-30 14:47:06	zhuzhr3	3A - Shortest path of the king	GNU C++17	Accepted	30 ms	3600 KB

3. Liebig's Barrels

You have m = n * k wooden staves. The i-th stave has length a_i. You have to assemble n barrels consisting of k staves each, you can use any k staves to construct a barrel. Each stave must belong to exactly one barrel.

Let volume v_j of barrel j be equal to the length of the minimal stave in it.



You want to assemble exactly n barrels with the maximal total sum of volumes. But you have to make them equal enough, so a difference between volumes of any pair of the resulting barrels must not exceed l, i.e.

$|v_x - v_y| \leq l$ for any $1 \leq x \leq n$ and 1<=y<=n.

Print maximal total sum of volumes of equal enough barrels or 0 if it's impossible to satisfy the condition above.

Input

The first line contains three space-separated integers n, k and l (1<=n, k<=10^5, 1<=n*k<=10^5, 1<=l<=10^9).

The second line contains m = n * k space-separated integers $a_1, a_2, \ldots, a_m$(1<= a_i<=10^9) — lengths of staves.

Output

Print single integer — maximal total sum of the volumes of barrels or 0 if it's impossible to construct exactly n barrels satisfying the condition $|v_x - v_y| \leq l$ for any 1<=x<=n and 1<=y<=n.

Examples

input
4 2 1
2 2 1 2 3 2 2 3
output
7

input
2 1 0
10 10
output
20

input
1 2 1
5 2
output
2

input
3 2 1
1 2 3 4 5 6
output
0

Note

In the first example you can form the following barrels: [1, 2], [2, 2], [2, 3], [2, 3].

In the second example you can form the following barrels: [10], [10].

In the third example you can form the following barrels: [2, 5].

In the fourth example difference between volumes of barrels in any partition is at least 2 so it is impossible to make barrels equal enough.

解题思路：

首先将 staves 数组排序，再判断是否可以组成符合条件的水桶，如果可以组成符合条件的，那么运用贪心算法找出可以组成的最大的木桶，然后将使用过的 stave 从数组中删除，接着再找可以组成的最大的木桶，如此下去直到数组为空。

代码如下：

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;
int main() {
    long long n, k, l, m, index, remain, volumes = 0;
    vector<unsigned long long> staves;
    cin >> n >> k >> l;
    m = n * k;
```

```
    for (unsigned long long i = 0; i < m; ++i) {
        unsigned long long tmp;
        cin >> tmp;
        staves.push_back(tmp);}
    sort(staves.begin(), staves.end());
    if (staves[n - 1] - staves[0] > l) {
        cout << 0;
        return 0;}
    index = m - 1;
    for (unsigned long long i = 0; i < m; ++i) {
        if (staves[i] - staves[0] > l) {
            index = i - 1;
            break;}}
    remain = m - 1 - index;
    while (remain != 0 && remain >= k - 1) {
        volumes += staves[index];
        index--;
        remain -= k - 1;}
    if (index < 0) {
        cout << volumes;
        return 0;}
    while (index > k - 1) {
        index -= k - 1 - remain;
        remain = 0;
        volumes += staves[index];
        index--;}
    volumes += staves[0];
    cout << volumes;
    return 0;}
```

结果：

38913634	2018-06-03 10:45:41	zhuzhr3	C - Liebig's Barrels	GNU C++17	Accepted	234 ms	4800 KB

4. Jamie and Interesting Graph

Jamie has recently found undirected weighted graphs with the following properties very interesting:

The graph is connected and contains exactly n vertices and m edges.

All edge weights are integers and are in range [1, 10^9] inclusive.

The length of shortest path from 1 to n is a prime number.

The sum of edges' weights in the minimum spanning tree (MST) of the graph is a prime number.

The graph contains no loops or multi-edges.

If you are not familiar with some terms from the statement you can find definitions of them in notes section.

Help Jamie construct any graph with given number of vertices and edges that is interesting!

Input

First line of input contains 2 integers n, m ($2<=n<=10^5$, $n-1<=m<=min((n(n-1)/2), 10^5)$) — the required number of vertices and edges.

Output

In the first line output 2 integers sp, mstw ($1<=sp, mstw<=10^{14}$) — the length of the shortest path and the sum of edges' weights in the minimum spanning tree.

In the next lines output the edges of the graph. In each line output 3 integers u, v, w ($1<=u,v<=n,1<=w<=10^9$) describing the edge connecting u and v and having weight w.

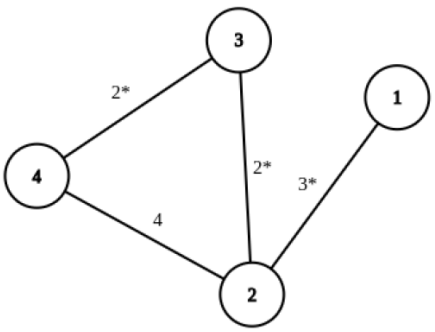
Examples

input
4 4
output
7 7
1 2 3
2 3 2
3 4 2
2 4 4

input
5 4
output
7 13
1 2 2
1 3 4
1 4 3
4 5 4

Note

The graph of sample 1:



Shortest path sequence: {1, 2, 3, 4}. MST edges are marked with an asterisk (*).

Definition of terms used in the problem statement:

A shortest path in an undirected graph is a sequence of vertices $(v_1, v_2, \cdots, v_k)$ such that v_i is adjacent to v_{i+1}(1<=i<=k) and the sum of weight $\sum_{i=1}^{n-1} w(v_i, v_{i+1})$ is minimized where w(i, j) is the edge weight between i and j.

A prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself.

A minimum spanning tree (MST) is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight.

解题思路：

从 n 开始找到一个最小的质数 p，然后构造一个最小生成树，树的形式为 1--2, 2--3, 3--4, … , n-1--n，前 n-2 条边的长度为 1，第 n-1 条边的长度为 p-(n-2)，最后随意构造 m-(n-1)条长度为 10^9 的边即可。

代码如下：

```
#include <iostream>
#include <cmath>
#include <vector>
using namespace std;
bool isPrime(long long num) {
    if (num == 2 || num == 3) return 1;
    if (num % 6 != 1 && num % 6 != 5) return 0;
    long long tmp = sqrt(num);
    for (long long i = 5; i <= tmp; i += 6)
        if (num % i == 0 || num % (i + 2) == 0)
            return 0;
    return 1;}
int main() {
    long long n, m, sp, mstw, remain;
    vector<long long> edges;
    cin >> n >> m;
    for (long long i = n;; i++) {
        if (isPrime(i)) {
            sp = mstw = i;
            break;}}
    for (long long i = 0; i < n - 2; ++i)
        edges.push_back(1);
    edges.push_back(mstw - n + 2);
```

```
remain = m - n + 1;
cout << sp << ' ' << mstw << endl;
for (long long i = 0; i < n - 1; ++i)
    cout << i + 1 << ' ' << i + 2 << ' ' << edges[i] << endl;
for (long long i = 1; i < n; ++i) {
    for (long long j = i + 2; j < n + 1; ++j) {
        if (remain == 0)
            return 0;
        cout << i << ' ' << j << ' ' << 1000000000 << endl;
        remain--;}}
return 0;}
```

结果：

38915573	2018-06-03 11:56:34	zhuzhr3	916C - Jamie and Interesting Graph	GNU C++17	Accepted	280 ms	4800 KB