```java
public static long func_c81196dc5a7e46378c2174d457e35a79(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return HI;
}

public static double func_5ac676344ff14ebb984ac0f3d16c1d95(int o,
int[] v, double[] l, double[] p){
    double best = 1e200;
    double lsum = 0, rsum = 0;
    double psum = 0;
    double m = 0;
    double q = 1;
    for (int i = 0; i < o; i++) {
        m += l[v[i]];
        psum += p[v[i]] * q;
        rsum += p[v[i]] * q * m;
        q *= 1 - p[v[i]];
    }
    m = 0;
    q = 1;
    return rsum;
}

public static int func_cb576104546a4cc0ad313b1b750a9a04(int i, int
high, int low, long rest, long[] partials){
    int guess = (low + high) / 2;
    long second = partials[guess] - partials[i];
    if (second * 2 > rest)
        high = guess;
    else
        low = guess;
    return high;
}

public static long[] func_0734f3e8317e422dbbcd9fd826914d1a(int N, long
p, long r, long s, long q){
```

```
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    return arr;
}

public static long[] func_6ccff2a304d94e76bda551522f39039d(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    return a;
}

public static long[] func_1bdf6922a9604d1dacb2876da3578d62(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
```

```
    return RS;
}

public static long func_5539237aea7b41939fed0a5304e3f4c6(int n, int x,
int y, long sum, long[] a){
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return tmp;
}

public static int func_d9de90aa334b4508bee10ab7e93a8474(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
```

```
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    return x;
}

public static int func_61166efb07c140ef98ad46e02aa8ed90(Scanner in){
    int w = in.nextInt();
    int ln = in.nextInt();
    int un = in.nextInt();
    int g = in.nextInt();
    C.l = new C.Tacka[ln];
    return g;
}

public static long[] func_95a2e3c3c6ee45b7aa4c3d22cbd7b4be(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
```

```
        }
    }
    double res = (A[N] - ans) * 1.0 / A[N];
    return A;
}

public static long[] func_31a279d1533e478788217e8d85a18f20(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return best;
}

public static long func_c3fe9af36ce0496fa3f62f79720a5a4d(int n, int j,
long prefixSum, long[] rsum){
    long sum = (rsum[j] - prefixSum);
    long suffSum = rsum[n - 1] - sum - prefixSum;
    long max = Math.max(sum, suffSum);
    return sum;
}
```

```
public static long[] func_8f364645674b4ad292421214443d2d2a(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    Locale.setDefault(Locale.US);
    return sum;
}
```

```java
public static double func_2f9a0c5034a24749847e7220c32710d9(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    return mx;
}

public static int[] func_93ed3cffbfe2438b99da66662a040ae8(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
        }
    }
    return qty;
}

public static long func_0921ad3bfb8b4ceb932f21976b7aa1f7(int lo, int
hi, int i, long left, long[] psum){
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
```

```
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    if (lo > i + 1) {
        t = psum[lo - 1] - psum[i];
        a = Math.min(a, Math.max(t, left - t));
    }
    return a;
}

public static int[] func_63f746ff16ad44eeae5eafe05f6e7b95(int u, int
l, int[] xs, int[] xu, int[] xl){
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    return areal;
}

public static long[] func_761c6498858a404ca65b9924b9f4e7a0(int r, int
s, int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    return a;
}

public static long func_d8f21b2e3777483cb9393592b9482e7d(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    return v1;
}

public static double func_7bd7aacae05e4edfae257830b2c4b172(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    return cy1;
}

public static int func_9bab837e1f394c7fb89ef069b120f124(int n, Scanner
input){
    int p = input.nextInt(), q = input.nextInt(), r = input.nextInt(),
s = input.nextInt();
```

```
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    return p;
}

public static long[] func_aa8456692e0a4bb3971caa0e7a7be028(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
```

```
    long min = best[N - 1];
    long subSum = 0;
    return best;
}

public static long func_c39cb1bda162469cb887c82c5b4c6b52(int
winningThings, long budget, long lowestBet, long payMoney, long[] x){
    for (int i = 0; i < winningThings; i++) {
        if (x[i] > lowestBet) {
            return -1;
        }
        payMoney += -x[i] + lowestBet;
    }
    for (int i = winningThings; i < x.length; i++) {
        if (x[i] <= lowestBet) {
            payMoney += lowestBet + 1 - x[i];
        }
    }
    if (payMoney > budget) {
        return -1;
    }
    long ourWin = 0;
    for (int i = 0; i < winningThings; i++) {
        ourWin += lowestBet - x[i];
    }
    return payMoney;
}

public static double func_c15f81a5f1704500b7bcaa1f47c434b7(double
xnew, double ku, double bu, double kl, double bl){
    double ynewl = kl * xnew + bl;
    double ynewu = ku * xnew + bu;
    return ynewu;
}

public static int func_1f836e7b9aca416585b63d9370e0f62c(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return second;
```

```
}

public static int[] func_fa6e3abd994d4805b777887c81b2af63(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    return a;
}

public static long[] func_118527a75c1843dfb897817e737e4d85(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return dp;
}

public static double func_4eaba7a2e66d4d6db28c3ab1d7707dcf(double
right, double dy1, double left, double dy2, double dx){
    double mid = (left + right) / 2.0;
    double dy2_ = (dy2 - dy1) * (mid / dx) + dy1;
    double S2 = (dy1 + dy2_) * mid / 2.0;
    return mid;
}
```

```java
public static long func_3edb7bb73c12402c9b79b640da3945c7(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return sum;
}

public static long[] func_92c3de8c7c664949a88f02048d107c0a(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    return partial;
}

public static long[] func_6ff81f22834148f8a3cbe59a674cbb0d(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return rsum;
}

public static double func_bdc37956673e4bcea2d05265102350a3(double l,
double s, double r, double mid, double curArea){
    if (s > curArea) {
        r = mid;
    } else {
        l = mid;
    }
    mid = (l + r) * .5;
    return mid;
```

```
}

public static double func_2b701737b3554fc18aeaa976800393bd(int y0, int
x0, int x, double[] low, Scanner in){
    int y = in.nextInt();
    double k = 1.0 * (y - y0) / (x - x0);
    for (int j = x0 + 1; j <= x; j++) {
        low[j] = k * (j - x0) + y0;
    }
    x0 = x;
    return k;
}

public static int[] func_a905884b61494cd1bbc9e00cee117549(Scanner sc){
    int w = sc.nextInt();
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    return xu;
}

public static long func_05cc8cdc0f2045c59ec8ef368fbdf762(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return t1;
}

public static int func_98474e6864204697ba1b73521f7dadb0(int caseNum,
```

```
PrintWriter pw, Scanner sc){
    System.out.println("Processing case " + caseNum);
    pw.println("Case #" + caseNum + ":");
    int W = sc.nextInt();
    int L = sc.nextInt();
    int U = sc.nextInt();
    int G = sc.nextInt();
    return U;
}

public static double func_e43d0bd814f24de682bfe726b72b215f(int r, int
s, int n, int p, int q){
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    return ret;
}

public static long func_fe3c12b054464e3896a82928df8162a1(int r, int s,
int p, int q, long curr){
    curr *= p;
    curr += q;
    curr %= r;
    curr += s;
    return curr;
}

public static int func_8b97d5e4aba9475889f4de8e0421fe1c(int n, long p,
long q, long s, long r){
    A.array = new long[n + 1];
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    return index;
}

public static int[] func_7bea171415bd4ca2ac88c7b58e87a597(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
```

```
    return a;
}

public static long[] func_f4896e6f36d74803a863510eb4c9df9f(int c, long
B, long[] left){
    long[] a = left.clone();
    long tmpB = B;
    for (int i = 0; i < c; i++) {
        tmpB -= a[c - 1] - a[i];
        a[i] = a[c - 1];
    }
    for (int i = c; i < A.C; i++) {
        if (a[i] == a[0]) {
            a[i]++;
            tmpB--;
        }
    }
    return a;
}

public static int func_b90c883b4b5242c080b6e32a45ac98ee(int rght, long
rs, long max, long[] pref){
    long sum = pref[rght];
    int left = -1, right = rght;
    while (left < right - 1) {
        int mid = (left + right) >> 1;
        if (pref[mid] * 2 >= sum) {
            right = mid;
        } else {
            left = mid;
        }
    }
    for (int t = right - 2; t <= right + 2; t++) {
        if (0 <= t && t <= rght) {
            long ans = rs;
            ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
            max = Math.min(max, ans);
        }
    }
    return right;
}

public static long func_01ea75053839402fb3cff5ccb2ffbc53(int n,
PrintWriter out, Scanner in){
    in.nextLine();
    for (int i = 0; i < n; i++) {
        out.printf("Case #%d: ", i + 1);
        new A(in, out).call();
        out.flush();
        if (A.INPUT.isEmpty())
```

```
            System.out.println("case " + (i + 1) + " solved.\t");
        }
        long end = System.currentTimeMillis();
        return end;
    }

    public static int func_29a0308964844b8b88c7b09021574aa7(int lo, int
    hi, int i, long left, long[] psum){
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
        return lo;
    }

    public static String func_16327118d85b4408b3bdefbf37f859f2(int n, long
    P, long S, long Q, long R){
        long[] a = new long[n];
        long ss = 0;
        for (int i = 0; i < n; i++) {
            a[i] = ((i * P + Q) % R + S);
            ss += a[i];
        }
        long l = 0;
        long r = ss;
        while (r > l + 1) {
            long m = (l + r) / 2;
            boolean ok = true;
            for (int i = 0; i < n; i++) {
                if (a[i] > m)
                    ok = false;
            }
            if (!ok) {
                l = m;
            } else {
                int c = 0;
                long q = 0;
                while (c < n && q + a[c] <= m) {
                    q += a[c];
                    c++;
                }
                q = 0;
                while (c < n && q + a[c] <= m) {
                    q += a[c];
                    c++;
                }
```

```
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return "" + 1.0 * (ss - r) / ss;
}

public static int func_3d681eb3865e4c39ab614758d2457560(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    return Lb;
}

public static long func_b9d50221bd4d4970a324633124b2d8d3(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
```

```
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return sum;
}

public static int func_00a8028780b64040aa078f92ec24a5af(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
```

```
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return index;
}

public static long func_b8c37c7040ec4ae3a9a309d34963695a(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    return v1;
}

public static double func_b70f2acbe0004db7bd72e2d2eefa2272(long t2,
long sum, long t1, long rest, double ans){
    double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) / sum;
    ans = Math.max(ans, pp);
    return pp;
}

public static int func_bbdf9ead137a4543b06e2245f16bd313(Scanner sc){
    int w = sc.nextInt();
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    return g;
}

public static long func_47a31bbecca341a3ad5edc06e4a57518(int r, int s,
int p, int q, long curr){
```

```
    curr *= p;
    curr += q;
    curr %= r;
    curr += s;
    return curr;
}

public static double func_cd918eee6334444199b217728b247c47(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return cy1;
}

public static int[] func_43541bbec04747a490ac1e1c5dd44b73(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    return t;
}

public static double func_0242033bcedd4b1eb08aeca9d289410c(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    return cy1;
}

public static double func_387feb36141f45c180e77a86ea75047d(int U, int
G, int L, Scanner input){
    CodeJam_Round3_A.lower = new double[L][2];
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    return g;
}
```

```java
public static long func_9d89793045e6413ab59ff08b346856f9(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    final long total = sums[N];
    long answer = 0L;
    return total;
}

public static long func_10e0483ea00f48d086bc959460f4325f(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return sum;
}

public static long func_07fa8c7cdc11453f86b5c09079ab52c7(int lo, int
N, int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    return t;
}

public static long func_04e4c7d3d83f45298d07f5f5771e4eba(int i, int n,
long s, long l, long[] a){
    long s2 = 0;
    for (int j = 0; j < n; j++) {
        if (j <= i) {
            s += (l - a[j]);
            s2 += (l - a[j]);
        } else if (a[j] < l + 1) {
```

```
            s += (l + 1 - a[j]);
        }
    }
    return s2;
}

public static int func_4c30fa5ae78b49658f0d817378cf2a15(int py, int
px, int i, int[][] pos, double[] bound){
    int nx = pos[i][0];
    int ny = pos[i][1];
    double div = nx - px;
    for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
        bound[j] -= (ny - py) / div * (j - px) + py;
    }
    px = nx;
    py = ny;
    return py;
}

public static long[] func_a24e951977aa4d7599a836b2b5b8f258(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
```

```
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return A;
}

public static double func_2c1766906a224970a19ac6adc566635d(int n){
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    return ret;
}

public static long func_9a52b0bf0c8e41c6938a6ddcda5c5ac0(int n, long
p, long q, long r, Scanner br){
    long s = br.nextInt();
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return s;
}

public static long func_2a506390bc6841d5b542a8e50e4b8d01(int r, int p,
int s, int q, int n){
```

```
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    long best = Long.MAX_VALUE;
    for (int i = 0; i < n; i++) {
        x += a[i];
        Long up = all.ceiling(x / 2 + 1);
        if (up != null) {
            long now = Math.max(up, Math.max(x - up, sum - x));
            best = Math.min(best, now);
        }
        Long down = all.floor(x / 2);
        if (down != null) {
            long now = Math.max(down, Math.max(x - down, sum - x));
            best = Math.min(best, now);
        }
        all.add(x);
    }
    return x;
}

public static long func_00b186c1a835492a9a22359c359a8600(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
```

```
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return sum;
}

public static int func_342cbffd208d414eb47598a1ecbf5162(int test){
    A.gtest = test;
    A.ans = 0;
    long B = A.nextLong();
    int n = A.nextInt();
    long[] left = new long[A.C];
    for (int i = 0; i < n; i++) {
        left[i] = A.nextLong();
    }
    return n;
}

public static int[] func_e65fa036174641da97940f5af5711a30(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    return a;
}

public static long[] func_60adc8726d9649b7bc4e12ed139cc5a0(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    return arr;
}

public static long func_8dbfb277c1fd445a87d9a0e5482151d1(int i, int n,
long s, long l, long[] a){
    long s2 = 0;
    for (int j = 0; j < n; j++) {
        if (j <= i) {
```

```
            s += (l - a[j]);
            s2 += (l - a[j]);
        } else if (a[j] < l + 1) {
            s += (l + 1 - a[j]);
        }
    }
    return s;
}

public static int func_f09a095a7cc1445ea877489265874540(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
```

```java
    }
    return index;
}

public static double func_8fb0e82b46a542318270e3e5d4152833(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    return mx;
}

public static double func_39e71df0a93f498486809407c6e792cb(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    return res;
}

public static int func_7dd1f3e054eb4981a430e6edf1fb63a4(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    return second;
}

public static long func_ab94c1bcc4b84b4f8a78ff1c43c377cb(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
```

```
    int j = 0;
    double ans = 0;
    for (int i = 0; i < n; i++) {
        while (j < n && t2 + a[j] <= rest - a[j]) {
            t2 += a[j];
            rest -= a[j];
            j++;
        }
        double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) /
sum;
        ans = Math.max(ans, pp);
        if (j < n) {
            pp = 1.0 * (sum - Math.max(t1, Math.max(t2 + a[j], rest -
a[j]))) / sum;
            ans = Math.max(ans, pp);
        }
        t1 += a[i];
        t2 -= a[i];
        if (j == i) {
            rest -= a[j];
            j++;
        }
    }
    return t1;
}

public static long[] func_3bbd9febf59f40ceb25eb353775f4a27(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return arr;
}
```

```
public static int func_6d6de5cda8264026a64d32ccb31948f9(int i, int
high, int low, long rest, long[] partials){
    int guess = (low + high) / 2;
    long second = partials[guess] - partials[i];
    if (second * 2 > rest)
        high = guess;
    else
        low = guess;
    return low;
}

public static long func_b7b4c4835d884daeba7c2b69540e7821(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return sum;
}

public static double func_b0dc6f87f86047fb94f87f61542521d1(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
```

```
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return answer;
}

public static int[] func_5fd65e059ea149b4a6fff927d594a5ee(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    return data;
}

public static double func_5e77a03ecdc044aca03cf6eb044b7bac(int n,
double X, double[] x, double[] y){
    double x0 = 0, y0 = 0;
    double total = 0;
    ArrayList<Double> lx = new ArrayList<Double>();
    ArrayList<Double> ly = new ArrayList<Double>();
    lx.add(0.0);
    ly.add(-2000.0);
    for (int i = 0; i < n; ++i) {
        if (x[i] > X + BaiA.eps) {
            double Y = (X - x[i - 1]) * (y[i] - y[i - 1]) / (x[i] -
x[i - 1]) + y[i - 1];
            lx.add(X);
            ly.add(Y);
            break;
        } else {
            lx.add(x[i]);
            ly.add(y[i]);
        }
    }
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    return x0;
}
```

```
public static double[] func_adf54763b15f452c846e3cc672b1e2e6(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    return upper;
}

public static long func_d33ad12a0a8447028cd86aa46798e182(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
```

```
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return subSum;
}

public static int func_ae9544faff47451b814ede713c405558(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    return x;
}

public static long func_6a6965e5db6b4025b2b6948eb7c4e519(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return r;
}
```

```java
public static long func_67094b47ea874e4882f6b7c6aa274f31(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return best;
}

public static int func_c5332175fad945cf971965bbed1095fc(long right,
long left){
    long mid = (left + right) / 2;
    int last = 0;
    boolean ok = false;
    long sum1 = 0;
    int first = 0;
    return last;
}

public static long[] func_057ae54752494ae29c3dd801584e2f33(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    return a;
}

public static long func_fc8581db2eac4c3187137281d194f12c(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    final long total = sums[N];
    long answer = 0L;
    return answer;
}
```

```java
public static double func_2cb6ce77a0b54f3983e3ef6791034969(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return ans;
}
```

```java
public static long[] func_bc4640b33a884d35b4fab34c5068058d(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return A;
}

public static double func_c5ee4a00dcf64f67a21b44bcc05b1d36(double yy,
double xx, double y0, double total, double x0){
    total += (x0 + xx) * (y0 - yy);
    x0 = xx;
    y0 = yy;
    return x0;
}

public static long func_4595ff61f6544ffda14748431558348d(int n, int r,
int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
```

```
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    return high;
}

public static long func_5d175e6399944882a1e798f9f7acea04(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    final long total = sums[N];
    long answer = 0L;
    return total;
}

public static int[] func_82f85a3e516f48bab0f806f94e1391a9(int r, int
p, int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return A;
}

public static long[] func_8af522d6b64342c08465c08b03a28acf(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
```

```
    // sout(a);
    long[] sum = new long[n + 1];
    return sum;
}

public static int[] func_f23d1e7ddf104b778cd753ff8a36b8ec(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    return qty;
}

public static long[] func_4a8541bf36904c3cbee7440304841580(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    return sums;
}

public static long func_b394b9a7ecfa49a6b98b89c7ac3ea5c0(int n, long
min, long subSum, long[] best, long[] A){
    subSum += A[n];
    min = Math.min(min, Math.max(subSum, best[n - 1]));
    return subSum;
}

public static long[] func_28df2a4d0b40403cb55b9f78cd5970c8(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
```

```
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    return arr;
}

public static long func_3923c11834d64eb59732541724f8fad1(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return hi;
}

public static double func_863cd41adc194d90b7becf6b7198c120(double l,
double s, double r, double mid, double curArea){
    if (s > curArea) {
        r = mid;
    } else {
        l = mid;
    }
    mid = (l + r) * .5;
    return mid;
}

public static long func_ed58aaaafb8b4770b07e72f1a2955a44(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
```

```
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return sum;
}

public static long[] func_2ea5abb678bb40329857475822d0e161(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return RS;
}

public static long func_10b2d78c2dd141099774863a5338632d(long sr, long
sm, long sl, long[] a){
    for (int i = 1; i < a.length; ++i) {
        sr += a[i];
    }
    long sum = sm + sr;
    long ans = Math.max(sm, sr);
    int l = 0;
    for (int r = 1; r < a.length; ++r) {
        sm += a[r];
        sr -= a[r];
        while (l < r && Math.max(sl + a[l], sm - a[l]) < Math.max(sl,
sm)) {
            sl += a[l];
            sm -= a[l];
            ++l;
        }
        ans = Math.min(ans, Math.max(sr, Math.max(sm, sl)));
    // System.out.println(" " + l + " " + r + " " + sl + " " + sm + "
" + sr + " " + ans);
    }
    return sr;
}
```

```java
public static int[] func_eb20a1a05b8d418da957b96535560b5a(int u, int
l, int[] xs, int[] xu, int[] xl){
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    return areal;
}

public static long func_aba5daae25e142598b1a6dbd2d14a0db(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    for (int i = 2, j = 1; i < n; i++) {
        while (j + 1 < i && TaskA.sum(0, j + 1) <= TaskA.sum(j + 1,
i)) {
            ++j;
        }
        minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i, n),
TaskA.sum(j, i)), TaskA.sum(0, j)));
        if (j + 1 < i) {
            minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i,
n), TaskA.sum(j + 1, i)), TaskA.sum(0, j + 1)));
        }
    }
    return minimal;
}

public static double func_1ec7aed6dcae48b6a6980900b7e22757(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return my1;
}

public static int[] func_11f3cefee4fc4e19a49bce4bb506585d(int p, int
q, int n, Scanner scanner){
    int r = scanner.nextInt();
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
```

```
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    return array;
}

public static int func_1bd7e483e6a046e09c2b0c9a1fa8c4eb(int t, Scanner
in){
    int w = in.nextInt();
    int l = in.nextInt();
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
```

```
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
    area /= g;
    return g;
}

public static long[] func_a3d66c0eed594de9a21cfe7d31d9ffc0(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    return sum;
}

public static int func_92f74d494a6645c1abd45862a0bb2432(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
```

```
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return index;
}

public static long func_1f1f26ec82f5468680d4cfb5bf88dbbb(int i, int
ans, int n, long max, long[] S){
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return left;
}

public static double func_360cfbdeab674eb89d807ec8a4034f64(double x,
A.Point p1, A.Point p2){
    double dy = p2.y - p1.y;
    double dx = p2.x - p1.x;
    return p1.y + (x - p1.x) * dy / dx;
}

public static int func_e24f27a6d2484ab6a48b63920f8db449(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    if (v1 < v2) {
        min = med1;
    } else {
        max = med2;
    }
    return max;
}

public static long[] func_19ade0095c724b2f87678e358e4030cf(int count,
int[] qty){
```

```
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
        }
    }
    return partial;
}

public static long[] func_4eccbb03e04f4fdb826b27071daa272e(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
```

```
        }
    }
    return cum;
}

public static long func_adc954f3873442d4924fc0e9f2db40cd(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    return avg;
}

public static int func_f54c29564f174810808e65a2a701ebaf(int lo, int i,
int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    return lo;
}

public static long func_e3044f33ab8a46a3b501bc0fc5cf2f03(int lo, int
N, int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
```

```
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    return a;
}

public static long func_8febaeca5a3a4ee7b162838d2f21207a(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    return ss;
}

public static long func_668587e8e3394ff2ab13a32c55894a7c(int n, long
mid, long[] a, long[] sum, boolean ok){
    long sum1 = 0;
    int first = 0;
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
        second--;
    }
    if (sum1 >= mid && sum2 >= mid) {
        if (second >= first) {
            ok = true;
        } else {
            long total = 0;
            if (second != -1)
                total += sum[second];
            if (first != n)
                total += sum[n - 1] - sum[first - 1];
            if (total >= mid)
                ok = true;
        }
    }
    return sum2;
```

```
}

public static int[] func_07b74b0292754f98b02f460997fecf3a(int l,
Scanner sc){
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    for (int i = 1; i < l; ++i) {
        areal[i] = areal[i - 1] + (xl[i] - xl[i - 1]) * (yl[i] + yl[i
- 1]);
    // System.out.format("  %d%n", areal[i]);
    }
    for (int i = 1; i < u; ++i) {
        areau[i] = areau[i - 1] + (xu[i] - xu[i - 1]) * (yu[i] + yu[i
- 1]);
    // System.out.format("  %d%n", areau[i]);
    }
    double[] ss = new double[l + u];
    double[] widths = new double[l + u];
    double[] areas = new double[l + u];
    for (int i = 0; i < l + u; ++i) {
        int il = 0;
        int iu = 0;
        while (il < l && xl[il] <= xs[i]) ++il;
        --il;
        while (iu < u && xu[iu] <= xs[i]) ++iu;
        --iu;
        // System.out.format("     %d (%d) %d (%d)%n", il, xl[il], iu,
xu[iu]);
        double a = 0;
        if (xu[iu] == xs[i]) {
            a += areau[iu];
        } else {
```

```
            double lambda = (xs[i] - xu[iu]) / ((double) (xu[iu + 1] -
xu[iu]));
            a += areau[iu] + (xs[i] - xu[iu]) * (2 * yu[iu] + lambda *
(yu[iu + 1] - yu[iu]));
        }
        // System.out.format("   %.6f%n", a);
        if (xl[il] == xs[i]) {
            a -= areal[il];
        } else {
            double lambda = (xs[i] - xl[il]) / ((double) (xl[il + 1] -
xl[il]));
            a -= areal[il] + (xs[i] - xl[il]) * (2 * yl[il] + lambda *
(yl[il + 1] - yl[il]));
        }
        areas[i] = a;
        if (xl[il] < xl[l - 1]) {
            ss[i] = (yu[iu + 1] - yu[iu]) / ((double) (xu[iu + 1] -
xu[iu])) - (yl[il + 1] - yl[il]) / ((double) (xl[il + 1] - xl[il]));
            widths[i] = yu[iu] + (yu[iu + 1] - yu[iu]) * (xs[i] -
xu[iu]) / ((double) (xu[iu + 1] - xu[iu]));
            widths[i] -= yl[il] + (yl[il + 1] - yl[il]) * (xs[i] -
xl[il]) / ((double) (xl[il + 1] - xl[il]));
        }
    }
    return xs;
}

public static double func_7a11cfdff87c4f159443c372e716fefc(int W, int
U, int G, int t, Scanner input){
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
```

```
                min = mid;
            }
            cuts[i] = mid;
        }
        System.out.println("Case #" + t + ":");
        return g;
    }

    public static long func_14242d41b57643909b3db2217b93e99e(int ans, int
    n, long max, long center, long[] S){
        long left = S[ans] - S[0];
        long right = S[n] - left - center;
        long take = S[n] - Math.max(center, Math.max(left, right));
        max = Math.max(max, take);
        return max;
    }

    public static Point func_6ae2777d86ec421eafd5e144ca03d4b8(double x,
    Segment upperSegment){
        Line vertical = new Point(x, 0).line(new Point(x, 1));
        Point upperIntersection = upperSegment.line().intersect(vertical);
        return upperIntersection;
    }

    public static long func_859bc12333d644b08a1f7e6995fb42c5(int n, int r,
    int l, long best, long[] s){
        long cur = 0;
        cur = Math.max(cur, s[l - 1]);
        cur = Math.max(cur, s[r] - s[l - 1]);
        cur = Math.max(cur, s[n] - s[r]);
        best = Math.min(best, cur);
        return best;
    }

    public static double func_2a0cb0c104614f47a4930cedf39214d8(int r, int
    s, int n, int p, int q){
        A.num = new long[n + 1];
        for (int i = 1; i <= n; i++) {
            long curr = i - 1;
            curr *= p;
            curr += q;
            curr %= r;
            curr += s;
            A.num[i] = A.num[i - 1] + curr;
        }
        double ret = 0;
        for (int i = 0; i < n; i++) {
            int min = i;
            int max = n - 1;
            while (max - min > 3) {
```

```
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    A.pw.println(ret);
    return ret;
}

public static double func_c805f7bb8a2a49879c1f49174546ce97(double l,
double s, double r, double mid, double curArea){
    if (s > curArea) {
        r = mid;
    } else {
        l = mid;
    }
    mid = (l + r) * .5;
    return mid;
}

public static long[] func_94903bf12ceb40d9aa81eb03f1398380(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    return amt;
}

public static double func_84b1a8338da64022aa6ecd8b917dd4a3(int m,
PrintWriter out){
    out.println("Case #" + (m + 1) + ":");
    double totalArea = A.area(0, A.W);
    double targetArea = totalArea / A.G;
    double curans = 0;
    for (int g = 0; g < A.G - 1; g++) {
        double low = curans;
```

```
        double high = A.W;
        while (low + 0.0000000001 < high) {
            double guess = (low + high) / 2;
            double a = A.area(curans, guess);
            if (a < targetArea) {
                low = guess;
            } else {
                high = guess;
            }
        }
        curans = (low + high) / 2;
        out.println(curans);
    }
    return curans;
}

public static double func_3c216515ca964887a92952dd3acce80a(long[] x){
    Arrays.sort(x);
    // System.out.println(Arrays.toString(x));
    double best = 0;
    return best;
}

public static long[] func_0b01ac23d4584a1698ad1559b6ca5406(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    return counts;
}

public static double func_d94c21cf27c94ceab0abd41a5de89873(double mid,
double min, double cutArea, double max){
    mid = (min + max) / 2.0;
    double a = CodeJam_Round3_A.evalArea(mid);
    // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea, a);
    if (a > cutArea)
        max = mid;
    else
        min = mid;
    return a;
}

public static long func_8e66e3ddccdd48c6b4f1288fcd3a4da4(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
```

```
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    return avg;
}

public static long[] func_99124dda47c5494ebe854762db8b10c3(int n, long
p, long q, Scanner br){
    long r = br.nextInt();
    long s = br.nextInt();
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    for (int i = 0; i < n; i++) {
        int low = i;
        int high = n - 1;
        long prefixSum = (i == 0 ? 0 : rsum[i - 1]);
        while (high - low > 1) {
            int mid = (high + low) / 2;
            long sum = (rsum[mid] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            if (sum <= suffSum) {
                low = mid;
            } else {
                high = mid;
            }
        }
        for (int j = low; j <= high; j++) {
            long sum = (rsum[j] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            long max = Math.max(sum, suffSum);
            max = Math.max(max, prefixSum);
            best = Math.max(best, rsum[n - 1] - max);
        }
    }
    return nums;
}
```

```java
public static int func_7e533c5a91e347519c51b30312385866(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    return first;
}

public static long func_a259c262ee064194b6575889c7f38c26(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return sum;
}

public static int[] func_51cdd56368a943ddae42dec615da3adb(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
```

```
    long[] sum = new long[n + 1];
    sum[0] = 0;
    return a;
}

public static long func_10f860adf1c5431a9c649d728e518c90(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return tot;
}

public static long func_33787946f2ea4b6ea9338920befb3c50(int r, int s,
int q, long curr){
    curr += q;
    curr %= r;
    curr += s;
    return curr;
}

public static long func_92afbeadb0dc486bb97e743c0dda0db3(int i, long
waste, long initadd, long[] v){
    int numtie = 0;
    for (int j = i; j < 37; j++) {
        if (v[j] == v[i]) {
            numtie++;
        }
    }
    if (v[i] == v[i - 1]) {
        v[i]++;
```

```
        initadd++;
        waste += numtie;
    }
    long bet = i * (v[i] - 1);
    return bet;
}

public static long func_c62cb6fd25b049f6a9748dd1d71b79fc(int q, int s,
int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
    // sout(i + " " + index);
    }
    return x;
}

public static long func_c8da6477071a4f1ab84bf410aef8b1b5(int n, long
p, long q, long r, Scanner in){
    long s = in.nextInt();
    A.array = new long[n + 1];
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    long min = Long.MAX_VALUE;
    return s;
}

public static double func_65ea5cf33aa94a91b607674aeb26aada(double y0,
double total, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    for (int i = 0; i < lx.size(); ++i) {
        double xx = lx.get(i);
        double yy = ly.get(i);
```

```
        total += (x0 + xx) * (y0 - yy);
        x0 = xx;
        y0 = yy;
    }
    return y0;
}

public static long func_3da018afc1094a2989920d4637f0d628(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return leftSum;
}

public static int func_9a9933056d2943049834d68878a29d6c(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
```

```
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return pow2;
}

public static int func_fcbe298782734094b6d56d117811456d(int n, long p,
long q, long s, long r){
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    return index;
}

public static int func_0735ad5595684870accd8c7e951d407f(A2.Pair p2,
A2.Pair p1){
    int x0 = p1.a;
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    return x0;
}

public static long[] func_75a52aa8832348c69d8b4ac3061ba305(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
```

```
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return A;
}

public static long func_28963037e45d4331b39f9eb05de89a83(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return sum;
}

public static long func_62b75fc3cffb4a52b3a0d28bfbf2d6fe(int
beforeHalf, int n, long sum, long beforeHalfSum, long[] A){
```

```
    while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum / 2)
{
        beforeHalfSum += A[beforeHalf];
        beforeHalf++;
    }
    long max = sum;
    if (beforeHalf > 0) {
        max = Math.max(beforeHalfSum, sum - beforeHalfSum);
    }
    return max;
}

public static int func_1e967dce28444123ba14acff96df07d9(int n, int p,
int q, int r, Scanner sc){
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
```

```
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return s;
}

public static int func_f023d43644af41a2bd317e9eaf12ee08(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    return first;
}

public static long func_fb8af7437abd401cb558bbb28851dccb(int lo, int
N, int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
```

```
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    if (lo > i + 1) {
        t = psum[lo - 1] - psum[i];
        a = Math.min(a, Math.max(t, left - t));
    }
    return t;
}

public static double func_169dbb2a01c446d7bd9b261071dad750(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    return t;
}

public static double func_1ee5d1a227c945c79bb3f465923e3d6c(double yy,
double xx, double y0, double total, double x0){
    total += (x0 + xx) * (y0 - yy);
    x0 = xx;
    y0 = yy;
    return y0;
}

public static long[] func_0748645126d740b181549af89990815a(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    return A;
}

public static double func_6d60f42957b84cd3aac6b16ee2e47012(int n,
double X, double[] x, double[] y){
    double x0 = 0, y0 = 0;
    double total = 0;
    ArrayList<Double> lx = new ArrayList<Double>();
    ArrayList<Double> ly = new ArrayList<Double>();
    lx.add(0.0);
    ly.add(-2000.0);
    for (int i = 0; i < n; ++i) {
        if (x[i] > X + BaiA.eps) {
            double Y = (X - x[i - 1]) * (y[i] - y[i - 1]) / (x[i] -
x[i - 1]) + y[i - 1];
```

```
            lx.add(X);
            ly.add(Y);
            break;
        } else {
            lx.add(x[i]);
            ly.add(y[i]);
        }
    }
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    return x0;
}

public static int[] func_4b8574edd9f740b9a1f194325921cabf(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    return qty;
}

public static long func_78a1a4f4431f44d8ac5990ca44c5fd03(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return lo;
```

```
}

public static double func_8fab02365620467fb63a1585f1456d44(double X,
double[] angleDown, double[] angleUp, double[] down, double[] up){
    double upY = (X - (int) X) * angleUp[(int) X + 1] + up[(int) X];
    double downY = (X - (int) X) * angleDown[(int) X + 1] + down[(int)
X];
    double l = X;
    double r = (int) X + 1;
    return l;
}

public static int func_1e9977bbc121431c989cbdc725327105(int n, long p,
long q, long s, long r){
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    long min = Long.MAX_VALUE;
    return index;
}

public static long[] func_093db6fac632412b9234fc8fd948a2bf(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
```

```
        }
    }
    return suffixes;
}

public static long func_4d5cbd5a606b4de498acb53c54aeaa9f(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return sum;
}

public static double func_0a04a25dc29d4cd59464bd8b16121b4c(int ulidx,
int llidx, double xe, double x1){
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    return d1;
}

public static long[] func_52f7c6e284a44740a13ab4e342f24613(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
```

```
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return A;
}

public static String func_ac4edab9b9c54eeb82afa2d54d665133(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
```

```
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return "" + 1.0 * (ss - r) / ss;
}

public static double func_ab5e6e855acd470f9a0c61709c7461d7(double cur,
double used, double b, double a, double req){
    double hi = 1.0;
    double lo = used;
    for (int j = 0; j < 100; ++j) {
        double mi = (hi + lo) / 2.0;
        double cand = ((b - a) * used + a + (b - a) * mi + a) * (mi -
used) / 2.0;
        if (cur + cand >= req) {
            hi = mi;
        } else {
            lo = mi;
        }
    }
    return hi;
}

public static long func_8e7a65c21c1c4e7e8e184bdcf911b5bf(int n, int[]
data, long lo, long hi){
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return hi;
}

public static double func_2a9c7db57e734486badfa4380d83a39d(double a,
```

```
double b, double c, double sss){
    c -= 2 * sss;
    double g = 0;
    if (Math.abs(a) >= 1e-11) {
        double discr = b * b - 4 * a * c;
        g = (-b + Math.sqrt(discr)) / (2 * a);
    } else {
        g = -c / b;
    }
    return g;
}

public static long[] func_4578c1f09e974b6f8acbdd05c698a2c2(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return arr;
}

public static double func_71d847dfcd5f4591b1c0c1a92fb065eb(int n,
double res, double right, double[] xs, double[] ys){
    for (int i = 0; i < n - 1; i++) {
        if (right < xs[i])
            continue;
        if (right < xs[i + 1]) {
            double a = (ys[i + 1] - ys[i]) / (xs[i + 1] - xs[i]);
            double b = ys[i] - a * xs[i];
            double y = a * right + b;
            res += (ys[i] + y) * (right - xs[i]) / 2;
        } else {
            res += (ys[i] + ys[i + 1]) * (xs[i + 1] - xs[i]) / 2;
        }
    }
    return res;
}

public static double func_4376a435650140148c8aaa2cd2caec5a(double t,
double k, double d1, double dx){
    double res = (t - d1) / k;
```

```
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    return temp;
}

public static int[] func_3a678cc98db44095a28867959f24f498(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    return t;
}

public static long[] func_def8855cc1694547bdb00cae69815a23(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    return A;
}

public static double func_c1bc18887a864e04a30c783442320cfa(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return mx;
}

public static double func_7c0eeb7c29604181ac87bff322fca813(int r, int
s, int n, int p, int q){
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
```

```
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    return ret;
}

public static int[] func_67ff5ccf89bf420cb8e15c352a2a9d27(int l, int
u, PrintWriter output, Scanner input){
    int g = input.nextInt();
    int[] lx = new int[l];
    int[] ly = new int[l];
    for (int i = 0; i < l; i++) {
        lx[i] = input.nextInt();
        ly[i] = input.nextInt();
    }
    int[] ux = new int[u];
    int[] uy = new int[u];
    for (int i = 0; i < u; i++) {
        ux[i] = input.nextInt();
        uy[i] = input.nextInt();
    }
    output.println();
    double cx = 0;
    return lx;
}

public static double func_eeea6c53e06b4b0c8554498d3edcd7f3(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
```

```
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    return ans;
}

public static double func_95705093d3e64a58b24f18da8bfa67d4(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
```

```
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return ans;
}

public static int func_12cd1c3266a94ba5ae3f1c0edf80676c(int dx, int
dy, int x, int y){
    int rx = x + dx;
    int ry = y + dy;
    return rx;
}

public static double func_c205804ea5634308b456e6b746ac9211(double y0,
double total, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    y0 = ly.get(ly.size() - 1);
    for (int i = 0; i < lx.size(); ++i) {
        double xx = lx.get(i);
        double yy = ly.get(i);
        total += (x0 + xx) * (y0 - yy);
        x0 = xx;
        y0 = yy;
    }
    return y0;
}

public static long func_a8c2fe90fb6442168014e9ceb9887662(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
```

```
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return hi;
}

public static double func_29121c49da47450dac9d4082a3e88808(int
winningThings, long budget, long lowestBet, long[] x){
    long payMoney = 0;
    for (int i = 0; i < winningThings; i++) {
        if (x[i] > lowestBet) {
            return -1;
        }
        payMoney += -x[i] + lowestBet;
    }
    for (int i = winningThings; i < x.length; i++) {
        if (x[i] <= lowestBet) {
            payMoney += lowestBet + 1 - x[i];
        }
    }
    if (payMoney > budget) {
        return -1;
    }
    long ourWin = 0;
    for (int i = 0; i < winningThings; i++) {
        ourWin += lowestBet - x[i];
    }
    return (double) ourWin / (double) winningThings * 36.0 - payMoney;
}

public static long[] func_4c94e1b2d8bf441ab6205d4d7011cf8c(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
```

```
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return arr;
}

public static int func_a99392762fbb41dabf9d7ce5c85a2543(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return first;
}

public static int func_911e7502a4474d7fa34af128ea39ab8b(int n, int p,
int q, int r, Scanner sc){
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
```

```
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return y;
}

public static double func_27d0e5d8117d43c4b4f8cca63e6a588d(double k,
double remain, double x, double y){
    double a = k;
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    return c;
}

public static int func_eea793bf17de4962a7a2f043cc4242de(int lo, int i,
int n, long max, long[] S){
    int hi = i;
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return ans;
}
```

```java
public static long func_5e4112ea39f44b60a3fed9c42d53e5b3(int c, long
tmpB, long[] a){
    for (int i = 0; i < c; i++) {
        tmpB -= a[c - 1] - a[i];
        a[i] = a[c - 1];
    }
    for (int i = c; i < A.C; i++) {
        if (a[i] == a[0]) {
            a[i]++;
            tmpB--;
        }
    }
    return tmpB;
}

public static long func_6f725202a8cd41dd831102eaa1c588d0(int n, long
p, long q, long s, long r){
    A.array = new long[n + 1];
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    long min = Long.MAX_VALUE;
    return min;
}

public static int[] func_5749ee781be6470199993bd18f8488bf(int l, int
u, int[] lax, int[] lay){
    for (int i = 0; i < l; i++) {
        lax[i] = A.sc.nextInt();
        lay[i] = A.sc.nextInt();
    }
    int[] uax = new int[u];
    return uax;
}

public static double func_655645c64b9b46b08d779a746a733449(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    return d;
}

public static String func_0d4efe6b690e4b44a86b447b695d247a(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
```

```
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return "" + 1.0 * (ss - r) / ss;
}

public static long func_f47ae7a22bd946d39b89b9d3b28c6cd7(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
```

```
    return minimal;
}

public static int func_ded7a1fa64674d1e917a0f085c155943(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    int Tba = 100 * Lb + La * (100 - Fb);
    return Fb;
}

public static long[] func_e6443b31c1c64e5a9a32776875b1b1ea(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return cum;
}

public static int[] func_533e9d37316548b882a6875d7f83d8d3(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
```

```
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return data;
}

public static long[] func_cb9ac42810dd4036bf821b0dc3f9b619(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    return a;
}

public static double func_79977d0b9ceb48a7ad5161ab96822a55(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    return c;
}

public static int[] func_f7ab6d57104744a596761d5344f51fda(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    return qty;
}

public static int func_dddc8a83a0b64554a682c1f72eef28fa(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
```

```
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return y;
}

public static long func_ff58dd7c7e8843449f24ba65191905a1(long x, long
best, long sum, Long down){
    long now = Math.max(down, Math.max(x - down, sum - x));
    best = Math.min(best, now);
    return now;
}

public static long func_7c402474e97e433da762637c106b1aac(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
```

```
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return R;
}

public static ArrayList func_b12880bdc9b441ec9c1039a873ac89a3(double
X, double y0, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    return lx;
}

public static long func_fa2eea7b286b46d18a7c47d0daef7325(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    return sum;
}

public static double func_9f378f3b93654f6a98b333d99d7f8642(int l, int
g, int u, int w, int[][] pos){
    double[] bound = new double[w + 1];
    int px = pos[0][0];
    int py = pos[0][1];
    for (int i = 1; i < l; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] -= (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    px = pos[l][0];
    py = pos[l][1];
    for (int i = l + 1; i < l + u; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
```

```
        for (int j = px + (i == l + 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] += (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    double sum = 0.0;
    for (int i = 0; i < w; ++i) {
        sum += (bound[i] + bound[i + 1]) / 2.0;
    }
    double req = sum / g;
    return req;
}

public static long[] func_9b59938a9af843ae9b7abc4e568dff76(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    long max = 0;
    return psum;
}

public static int func_f1ef946ab3e646f7a0fca31355d6c525(int l, Scanner
sc){
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    return g;
}

public static int[] func_a3dd3f94853d4b568021ac1ec26accc2(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
```

```
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    return a;
}

public static double func_874eb4b29bef48ca904b73576eed41c8(double X,
double[] angleDown, double[] angleUp, double[] down, double[] up){
    double upY = (X - (int) X) * angleUp[(int) X + 1] + up[(int) X];
    double downY = (X - (int) X) * angleDown[(int) X + 1] + down[(int)
X];
    double l = X;
    return downY;
}

public static long[] func_981b41a01d7e4b05bd759585699e8a2a(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    return a;
}

public static int func_354a9d4aace74c7fbd7754c36b7d27e6(int n, int x,
int y, long sum, long[] a){
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
```

```
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return x;
}

public static int func_2b7930b22afa404cbf1ce9114e63f9f9(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
```

```
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return index;
}

public static long func_9d46d29fc2454ebf9f903fe4531c128b(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    return res;
}

public static int func_48a9ef652a4849918e2b25ef91663566(int N, long q,
long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    return beforeHalf;
}

public static double func_93c4a38026bd4080954a0a7003648b68(int r, int
s, int n, int p, int q){
    A.num = new long[n + 1];
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
```

```
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    return ret;
}

public static long[] func_401c39cfcbb74872928f2d69e190cbc5(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    return sum;
}

public static long func_e0728a6e394c4c52b1e7dd5c33d311f9(int N,
Scanner in){
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] d = new long[N];
    return r;
}

public static long[] func_9dc42f8b900544b68f8584a49c6a5ed5(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
```

```
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return dp;
}

public static long func_50c5f6c87547421fa124b0b35df34ecf(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
```

```
    }
    return min;
}

public static int[] func_5e922e248b23468b9f2cb75a8ae8ffbc(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    return ps;
}

public static TreeSet func_3b132daee07949f6bb717fd8b8ae1efa(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    return all;
}

public static double func_4b45c574185e41d9afca153c24f090c1(int l,
double ans, double ax, double right){
    double bx = Math.min(A.lowx[l + 1], right);
    double ay = A.lowy[l];
    double by = A.lowy[l + 1];
    if (ax > A.lowx[l]) {
        ay = A.lowy[l] + (ax - A.lowx[l]) / (A.lowx[l + 1] -
A.lowx[l]) * (A.lowy[l + 1] - A.lowy[l]);
    }
    if (bx < A.lowx[l + 1]) {
        by = A.lowy[l + 1] + (bx - A.lowx[l + 1]) / (A.lowx[l] -
A.lowx[l + 1]) * (A.lowy[l] - A.lowy[l + 1]);
    }
    ans -= (bx - ax) * ((ay + by) / 2 + 1000);
    return bx;
}

public static double func_a41dbe79b50b4d82ac13d9f5c4636294(double cx,
double mx, double ca, double la){
    cx = mx;
    ca -= la;
    return cx;
}
```

```java
public static long func_feb174bdcc7e4453a98290985def73ca(int n, int c,
long m, long q, long[] a){
    q = 0;
    while (c < n && q + a[c] <= m) {
        q += a[c];
        c++;
    }
    return q;
}

public static long[] func_eb4a962861334c9c9c57284b7b0e925f(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    return sum;
}

public static long[] func_fe3efe7c75b14e1e8077aaaed6fa837c(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    return suffixes;
}

public static int func_3bbd4cd66d2b4c1abe6074b957b9f157(int lo, int i,
int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
```

```
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    return ans;
}

public static int[] func_7ba5a39d18184d84bfa528747596cf6b(int L, int
U){
    int[] lx = new int[L], ly = new int[L];
    int[] ux = new int[U], uy = new int[U];
    return uy;
}

public static long func_b18e9707cc0e498281e6a5553139e00b(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return min;
}

public static int[] func_4aadb73d72b44551bdd324c9ad3ee5df(int u, int
l, Scanner sc){
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    return xu;
}

public static int func_37c0bda6761a4ed183b0336ace0739cc(int t, int w,
int u, int l, Scanner in){
```

```
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    return x0;
}

public static long func_3019179bd55d4b8eb6868d3871ee6541(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return rest;
}

public static double func_e24aa69c41e84b17ac79fc23c23708d4(int W, int
U, int G, int L, Scanner input){
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
```

```
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    return g;
}

public static int func_28b9d1ff95d24fe68947bcdee8a5e02f(int min, int
h, int max, long[] imos){
    int med1 = (min * 2 + max) / 3;
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    if (v1 < v2) {
        min = med1;
    } else {
        max = med2;
    }
    return med1;
}

public static long func_caab70f33d9d4bca8f4a82d5855bd16e(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    return answer;
}
```

```
public static long func_763844468219462f9d527826d4de4995(int
winningThings, long high, long low, long budget, long[] x){
    long mid = low + high >> 1;
    if (A.canDo(budget, x, winningThings, mid)) {
        low = mid;
    } else {
        high = mid;
    }
    return low;
}

public static long func_b9ab29869c6e4c47ac1a2cfc73e5860d(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return r;
}

public static long func_150be9fa7e6e4e6196f64954780d1560(int n, long
b, long s, long[] a, double max){
    s += a[n];
    double s2 = 0;
    for (int l = 0; l <= n; l++) {
        s2 += a[l];
        long x = (b - (n - l) + s) / (n + 1);
        if (x >= a[l] && (l + 1 >= a.length || x + 1 >= a[n]) && (n +
1 >= a.length || x < a[n + 1])) {
            if (max < 36 * (x - s2 / (l + 1)) - (n + 1) * x + s - (n -
l)) {
                max = 36 * (x - s2 / (l + 1)) - (n + 1) * x + s - (n -
l);
            }
        }
    }
    return s;
}

public static long[] func_43c9a77dc601460db58a4e575d2f5852(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
```

```
    int beforeHalf = 0;
    return A;
}

public static double func_a65844cb223e4fc981881415053d60d6(int il, int
iu, double xnew, double[][] u, double[][] l){
    double kl = (l[il + 1][1] - l[il][1]) / (l[il + 1][0] - l[il][0]);
    double bl = l[il][1] - (kl * l[il][0]);
    double ku = (u[iu + 1][1] - u[iu][1]) / (u[iu + 1][0] - u[iu][0]);
    double bu = u[iu][1] - (ku * u[iu][0]);
    double ynewl = kl * xnew + bl;
    double ynewu = ku * xnew + bu;
    return ku;
}

public static int[] func_c7adf1c2606147e3970878b3ee8129a9(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    return a;
}

public static double func_be5b1672938740ddb1fef2d613095b2f(int width,
int j, long mh, long all, double exp){
    long ma = Math.min(mh, (A.B - all - j) / width);
    exp += ma * 36.0;
    return exp;
}

public static long func_0569706e75f14cbfa4f83218fe0ec981(int minCol,
long level, long[] bets){
    if (level <= 0) {
        return -1;
    }
    long need1 = 0;
    for (int i = 0; i < minCol; i++) {
        if (bets[i] > level) {
            return -1;
        }
        need1 += level - bets[i];
    }
    return need1;
}
```

```java
public static long[] func_d52f4c6cf323424ba88ff38382f1a567(int N,
Scanner in){
    long p = in.nextInt();
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    return best;
}

public static int func_4c0f922f29054682bb0916c7c7ff177d(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return first;
}

public static double func_fb3a18c5717640498af5de66419e421f(int r, int
s, int n, int p, int q){
    A.num = new long[n + 1];
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
```

```
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    return ret;
}

public static long func_151fbb4c80144aa6bf1eeddd0ce162ca(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    return LO;
}

public static long[] func_b8fe0de9ef4e436f8a9aed3b55dd7af5(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    for (int i = 0; i < n; i++) {
        int low = i;
        int high = n - 1;
        long prefixSum = (i == 0 ? 0 : rsum[i - 1]);
        while (high - low > 1) {
            int mid = (high + low) / 2;
            long sum = (rsum[mid] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
```

```
            if (sum <= suffSum) {
                low = mid;
            } else {
                high = mid;
            }
        }
        for (int j = low; j <= high; j++) {
            long sum = (rsum[j] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            long max = Math.max(sum, suffSum);
            max = Math.max(max, prefixSum);
            best = Math.max(best, rsum[n - 1] - max);
        }
    }
    return nums;
}

public static int func_dfebe504beb14426b95f5cdb1921aba9(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
```

```
    }
    return beforeHalf;
}

public static long func_3726352e72054c5eb56211c67bd77d90(int min, int
h, int max, long[] imos){
    int med1 = (min * 2 + max) / 3;
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    return v1;
}

public static long func_6273ffb848dc464bbcf24a2a3a13ad0e(int j, int N,
int i, long ans, long[] A){
    long third = A[N] - A[i + 1];
    while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
    {
        long first = A[j + 1];
        long second = A[i + 1] - A[j + 1];
        ans = Math.min(ans, Math.max(first, Math.max(second, third)));
    }
    return ans;
}

public static double func_7a1209d3025a40e28972a97c0c1e28aa(int t, int
w, int g, double[] len, PrintWriter out){
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
    area /= g;
    out.println("Case #" + (t + 1) + ":");
    double x = 0;
    return x;
}

public static int func_50eec5c3fc9b4310bbf0d5b593ead38b(int n, int x,
int y, long sum, long[] a){
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    return x;
```

```
}

public static long func_b9f2a5ed64ab4779b319103014953edd(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalfSum;
}

public static int func_e9b04dbee4da47ddafa28de12dedd615(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
```

```
            tmp += a[i];
            if (tmp * 3 <= sum) {
                x = i;
            }
            if (tmp * 3 <= sum * 2) {
                y = i;
            }
        }
        double ans = 0;
        for (int dx = -3; dx <= 3; dx++) {
            for (int dy = -3; dy <= 3; dy++) {
                int rx = x + dx;
                int ry = y + dy;
                if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                    continue;
                }
                long sum1 = 0, sum2 = 0, sum3 = 0;
                for (int i = 0; i < n; i++) {
                    if (i < rx) {
                        sum1 += a[i];
                    } else {
                        if (i <= ry) {
                            sum2 += a[i];
                        } else {
                            sum3 += a[i];
                        }
                    }
                }
                long maxSum = Math.max(sum1, Math.max(sum2, sum3));
                double rate = (sum - maxSum) * 1.0 / sum;
                if (rate > ans) {
                    ans = rate;
                }
            // out.write(maxSum + " " + sum + "\n");
            }
        }
        return y;
    }

    public static long func_2ac2e28f1649484abafebee1594b27ff(int r, int s,
    int n, int p, int q){
        long sum = 0;
        long[] a = new long[n];
        for (int i = 0; i < n; i++) {
            a[i] = (((long) i * p + q) % r + s);
            sum += a[i];
        }
        long min = sum;
        long si = 0, sj = 0;
        for (int i = 0, j = n - 1; i <= j; ) {
```

```
        if (si + a[i] < sj + a[j]) {
            si += a[i++];
        } else {
            sj += a[j--];
        }
        min = Math.min(min, Tour.max(sum - si - sj, si, sj));
    }
    return si;
}

public static int func_146ed11395394b28bc6a3774e9619f99(int i, int x,
int y, long tmp, long sum){
    if (tmp * 3 <= sum) {
        x = i;
    }
    if (tmp * 3 <= sum * 2) {
        y = i;
    }
    return x;
}

public static long func_f69e20b4749345ec9f17123944efb327(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return avg;
}

public static long[] func_a2f91011e82c44dd9ff0b7f89d4ebed6(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    long max = 0;
```

```java
    return psum;
}

public static long func_dd8059afcd674e62b3aae09663bfecdc(int p, int q,
int n, int r, int s){
    TaskA.a = new long[n];
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    return minimal;
}

public static double func_fab4f0fab41945c1b23936ee2708531b(double
chunkLeft, double thisLow, double currentLow, double thisHigh, double
currentHigh){
    currentHigh = thisHigh;
    currentLow = thisLow;
    chunkLeft = 0;
    return chunkLeft;
}

public static int func_d2d431fd58404024bf88e91dd4a3b50a(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    return g;
}

public static double func_549887b903f448e4b6c5276f218eca6c(double dg,
double s, double lg, double p){
    double tp = C.povrsinaDo(s);
    if (tp < p)
        lg = s;
    else
        dg = s;
    return dg;
}
```

```java
public static long[] func_deb872ebdc8a445880ea0696fc384b7d(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    return arr;
}

public static long func_2b4ab4714f954f9c9c9ddb281e54a760(int s, int N,
int q, int p, int r){
    long[] n = new long[N];
    for (int i = 0; i < N; i++) {
        n[i] = ((long) i * p + q) % r + s;
        if (i > 0) {
            n[i] += n[i - 1];
        }
    }
    long ans = Long.MAX_VALUE / 4;
    return ans;
}

public static double func_59c3d13141474de8b042b7a9bb10744e(double r,
double l, double last, double ss){
    last = (l + r) / 2;
    ss = 0;
    return last;
}

public static double func_a68e5713e5ae4d808355d70d768528f1(double mid,
double min, double cutArea, double max){
    mid = (min + max) / 2.0;
    double a = CodeJam_Round3_A.evalArea(mid);
    // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea, a);
    if (a > cutArea)
        max = mid;
    else
        min = mid;
    return max;
}

public static long func_043399fbcb894027bc30f80a7051787e(int n, int p,
int q, int r, int s){
```

```
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return sum;
}

public static long[] func_545d57c7200d49f8b5bd71ed0e8d932f(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return cum;
```

```
}

public static int[] func_4a0dc758344447a282ebd646c990669a(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    return a;
}

public static long func_8b01313abfc6414180bfbbb72f9889b8(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    for (int i = 0; i < n; i++) {
        while (j < n && t2 + a[j] <= rest - a[j]) {
            t2 += a[j];
            rest -= a[j];
            j++;
        }
        double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) /
sum;
        ans = Math.max(ans, pp);
        if (j < n) {
            pp = 1.0 * (sum - Math.max(t1, Math.max(t2 + a[j], rest -
a[j]))) / sum;
            ans = Math.max(ans, pp);
        }
        t1 += a[i];
        t2 -= a[i];
        if (j == i) {
            rest -= a[j];
            j++;
        }
    }
    return t2;
```

```
}

public static long[] func_022369e9c9e94c348e7455789c732058(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    return suffixes;
}

public static double func_e014c3c98893470ab16df502ac1e40b6(double
target, double hi, double lastx, double lo){
    double mid = 0;
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    return hi;
}

public static long[] func_dc5f4fb05df64d10ba495ce061bdf456(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    return arr;
}

public static int func_f93f718f34da456b97cba0a0571fa5a2(int n, long p,
long q, long s, long r){
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    return index;
```

```
}

public static double func_e0ba947c77044712adedc3266e81b763(int
winningThings, long budget, long lowestBet, long payMoney, long[] x){
    for (int i = 0; i < winningThings; i++) {
        if (x[i] > lowestBet) {
            return -1;
        }
        payMoney += -x[i] + lowestBet;
    }
    for (int i = winningThings; i < x.length; i++) {
        if (x[i] <= lowestBet) {
            payMoney += lowestBet + 1 - x[i];
        }
    }
    if (payMoney > budget) {
        return -1;
    }
    long ourWin = 0;
    for (int i = 0; i < winningThings; i++) {
        ourWin += lowestBet - x[i];
    }
    return (double) ourWin / (double) winningThings * 36.0 - payMoney;
}

public static double func_3cc54b6358374cd28c0425257b76a093(double
target, double hi, double mid, double lastx, double lo){
    mid = (hi + lo) / 2;
    double ar = A.area(lastx, mid);
    if (ar > target)
        hi = mid;
    else
        lo = mid;
    return hi;
}

public static long func_5eb9cf2ef76a442483b8d610045df6a2(int n, int b,
int a, int i, long[] sum){
    long s1 = sum[b - i + 1] - sum[a];
    long s2 = sum[a];
    long s3 = sum[n] - sum[b - i + 1];
    return s3;
}

public static int func_604d958e3d5c4497b54d83727865a00b(int r, int p,
int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
```

```
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    return s;
}

public static int func_feefe796460146edb178547e20d8183e(int i, long[]
a){
    long now = a[i - 1];
    long eq = 0;
    for (int j = 0; j < i; j++) {
        eq += now - a[j];
    }
    int same = 0;
    return same;
}

public static int func_67326d76b7ff43d1af01bf364061eb10(int u, int n,
int l, int[] xl, int[] xu){
    int[] x = new int[n];
    for (int i = 0; i < u; i++) x[i] = xu[i];
    for (int i = 1; i < l - 1; i++) x[i - 1 + u] = xl[i];
    Arrays.sort(x);
    double[] y1 = new double[n];
    int ind = 0;
    return ind;
}

public static long func_793b80a420044b72b21d891d77cf681a(long sr, long
sm, long[] a){
    for (int i = 1; i < a.length; ++i) {
        sr += a[i];
    }
    long sum = sm + sr;
    long ans = Math.max(sm, sr);
    return ans;
}

public static long func_355e8e829dc249189a524620245b178c(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
```

```
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return LO;
}

public static long[] func_82eed32a3efc4b0f8756c7b1b746c6e0(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return arr;
}

public static int func_3a66ff02d6c04e6b9315f49d6ba65b83(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
```

```
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return index;
}

public static long func_aebdfeee7fb9401b8926b45452d8b94a(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return tot;
}

public static long func_182334a0cb15466dbd5fedc95b3af273(int N, long
q, long p, long r, long s){
```

```
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return min;
}

public static long func_3f17dcb28f9d4c8aafcf55536989de71(int N, int[]
arr, long sum, long[] LS){
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    while (pow2 * 2 <= N) {
        pow2 *= 2;
    }
    return LO;
```

```
}

public static double func_3ae93dc7d8da4939b8cfbe25299785fc(int p2, int
p1, int[] y2, int[] y1, int[] x1){
    double a1 = y1[p1 + 1] - y1[p1];
    double b1 = x1[p1] - x1[p1 + 1];
    double c1 = a1 * x1[p1] + b1 * y1[p1];
    double a2 = y2[p2 + 1] - y2[p2];
    return c1;
}

public static long func_45992a60d3384ee994e4c99dafabd44f(int i, int
ans, int n, long diff1, long[] S){
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    return diff2;
}

public static long func_66010b8c2c7a4c4c95ca3ca1a6974deb(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    while (pow2 * 2 <= N) {
        pow2 *= 2;
    }
    return sum;
}

public static double func_792cba37b6114973bed13ffe511b5491(int i, long
```

```
v1, long bet, long waste){
    double val = v1 / (i + 0.0);
    val -= bet;
    val -= waste;
    System.out.println(val);
    return val;
}

public static long[] func_552f5754f8b2466ca6053824a2164853(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    long max = 0;
    return psum;
}

public static long func_4a0760540a4d43a49f2341cadf29cf6e(int n, int p,
int q, Scanner sc){
    int r = sc.nextInt();
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return tmp;
}

public static int func_1cce610bf8c44984bf708aca3d8d1f2b(int t, int w,
int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
```

```
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    return y0;
}

public static long func_c68823f418cd45bb9b085b82f77865f9(int
winningThings, long budget, long lowestBet, long[] x){
    long payMoney = 0;
    for (int i = 0; i < winningThings; i++) {
        if (x[i] > lowestBet) {
            return -1;
        }
        payMoney += -x[i] + lowestBet;
    }
    for (int i = winningThings; i < x.length; i++) {
        if (x[i] <= lowestBet) {
            payMoney += lowestBet + 1 - x[i];
        }
    }
    if (payMoney > budget) {
        return -1;
    }
    return payMoney;
}

public static long[] func_08e0b57adc654ec8b2b92b40a4ef1fed(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    return nums;
}

public static int func_02c0ea6b03b74d0c9d2b3882465d7250(int N, int
pow2, long req, long[] arr){
    int index = 0;
```

```
    for (; pow2 > 0; pow2 /= 2) {
        if (index + pow2 >= N) {
            continue;
        }
        if (arr[index + pow2] <= req) {
            index += pow2;
        }
    }
    return pow2;
}

public static long[] func_08f9285bd8de4f62af2f64778ef46948(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
```

```
                l = m;
            }
        }
    }
    return a;
}

public static long func_841188e67f244f31baaca5e36b5c1ced(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return s2;
}

public static long func_7285792cb54f436e9bd3856d02e76aca(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
```

```
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return min;
}

public static double[] func_0fba2d65581d402aa3b0466445102925(int t,
int w, int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
```

```
    return len;
}

public static long func_bbc79bd82841433b8b463e4cd680bf43(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    return minimal;
}

public static long func_728b7e5708b1440380ad6d28233acf05(int n, long
prefixSum, long sum, long best, long[] rsum){
    long suffSum = rsum[n - 1] - sum - prefixSum;
    long max = Math.max(sum, suffSum);
    max = Math.max(max, prefixSum);
    best = Math.max(best, rsum[n - 1] - max);
    return max;
}

public static long func_46b3662f4462415797a49504fcb301d9(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return s2;
}

public static int func_0dbdef5ba11343e7bcfa3fe890d6ed0d(int l, Scanner
input){
    int u = input.nextInt();
    int g = input.nextInt();
    int[] lx = new int[l];
    int[] ly = new int[l];
    for (int i = 0; i < l; i++) {
        lx[i] = input.nextInt();
        ly[i] = input.nextInt();
    }
    int[] ux = new int[u];
    int[] uy = new int[u];
    for (int i = 0; i < u; i++) {
```

```
        ux[i] = input.nextInt();
        uy[i] = input.nextInt();
    }
    return u;
}

public static long[] func_e154584a327b406e8d75fcc67b12d085(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    return a;
}

public static int func_d69df39b371d4625bcd7af8605dfbe81(int N, int i,
long[] psum){
    int lo = i + 1, hi = N;
    long left = psum[N] - psum[i];
    return hi;
}

public static double func_e1f722f89f034cf8884d1947583681d3(int
taskIndex, int N, long sum, long[] arr, long[] dp){
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
```

```
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    System.out.println(String.format("Case #%d: %.11f", taskIndex,
res));
    return res;
}

public static int func_2f1a659e39dc4de89a77de4740662cf5(int t, Scanner
in){
    int w = in.nextInt();
    int l = in.nextInt();
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    return y0;
}

public static long func_ebb430db211f40dd84b5172815c78e1c(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
```

```
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return beforeHalfSum;
}

public static double func_8dee98008f824f9ead9f6932a1151efe(int K, long
use, long req, long[] arr, double pay){
    for (int i = 0; i < K; ++i) {
        pay += (req - arr[i]);
    }
    pay *= 36;
    pay /= K;
    return pay - use;
}

public static double func_8b8abdc749884ed4ba34d1483765832b(int
casenumber, int l, Scanner sc){
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    for (int i = 1; i < l; ++i) {
        areal[i] = areal[i - 1] + (xl[i] - xl[i - 1]) * (yl[i] + yl[i
- 1]);
    // System.out.format("  %d%n", areal[i]);
    }
    for (int i = 1; i < u; ++i) {
        areau[i] = areau[i - 1] + (xu[i] - xu[i - 1]) * (yu[i] + yu[i
- 1]);
```

```
    // System.out.format("  %d%n", areau[i]);
    }
    double[] ss = new double[l + u];
    double[] widths = new double[l + u];
    double[] areas = new double[l + u];
    for (int i = 0; i < l + u; ++i) {
        int il = 0;
        int iu = 0;
        while (il < l && xl[il] <= xs[i]) ++il;
        --il;
        while (iu < u && xu[iu] <= xs[i]) ++iu;
        --iu;
        // System.out.format("     %d (%d) %d (%d)%n", il, xl[il], iu,
xu[iu]);
        double a = 0;
        if (xu[iu] == xs[i]) {
            a += areau[iu];
        } else {
            double lambda = (xs[i] - xu[iu]) / ((double) (xu[iu + 1] -
xu[iu]));
            a += areau[iu] + (xs[i] - xu[iu]) * (2 * yu[iu] + lambda *
(yu[iu + 1] - yu[iu]));
        }
        // System.out.format("   %.6f%n", a);
        if (xl[il] == xs[i]) {
            a -= areal[il];
        } else {
            double lambda = (xs[i] - xl[il]) / ((double) (xl[il + 1] -
xl[il]));
            a -= areal[il] + (xs[i] - xl[il]) * (2 * yl[il] + lambda *
(yl[il + 1] - yl[il]));
        }
        areas[i] = a;
        if (xl[il] < xl[l - 1]) {
            ss[i] = (yu[iu + 1] - yu[iu]) / ((double) (xu[iu + 1] -
xu[iu])) - (yl[il + 1] - yl[il]) / ((double) (xl[il + 1] - xl[il]));
            widths[i] = yu[iu] + (yu[iu + 1] - yu[iu]) * (xs[i] -
xu[iu]) / ((double) (xu[iu + 1] - xu[iu]));
            widths[i] -= yl[il] + (yl[il + 1] - yl[il]) * (xs[i] -
xl[il]) / ((double) (xl[il + 1] - xl[il]));
        }
    }
    System.out.format("Case #%d:%n", casenumber);
    double area = areau[u - 1] - areal[l - 1];
    return area;
}

public static int[] func_9438d194e63a449490d5ae1ea7b6a0e5(int t,
Scanner input){
    System.out.printf("Case #%d: ", t + 1);
```

```
    int n = input.nextInt();
    int p = input.nextInt(), q = input.nextInt(), r = input.nextInt(),
s = input.nextInt();
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    return data;
}

public static long[] func_81ae1800bde346e3b08745854850de2c(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return arr;
}

public static double func_905a50007b264bfaa6811a0feef68ebd(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
```

```
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    return res;
}

public static double func_c19d351b35c64c86bce7bbc23ae97c34(int i,
double[] y, double[] x, double[] angleUp, double[] up){
    double Y = (y[i] - y[i - 1]) / (x[i] - x[i - 1]);
    double z = y[i - 1] + Y;
    for (int j = (int) x[i - 1] + 1; j <= x[i]; j++) {
        up[j] = z;
        z += Y;
        angleUp[j] = Y;
    }
    return Y;
}

public static int[] func_9ee8a4cf13fc40118bcad3cb5fe28432(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    return data;
}

public static int[] func_345fb614e3b54f659aec681d5105df98(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
```

```
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return arr;
}

public static long[] func_d83b461b79304fbea9de2fe642c764f7(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
```

```
            }
        }
    }
    return a;
}

public static long func_3ebf2058691044e6bd5c64f9303ab5e3(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return L;
}

public static long func_67cbd6996283424d912cd5c02df84997(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return best;
}
```

```java
public static long func_1f8f1f7ccdca46739e321fae29c32307(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return temp;
}

public static long func_625549bc3d804ae78626e8618db00716(int n, int r,
int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    return high;
}

public static long func_40cca305705b4fee827786aa0894f668(int N, int
beforeHalf, long beforeHalfSum, long[] best, long[] A){
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
```

```
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return sum;
}

public static long[] func_92f406ec1548478ebc96164c74d08074(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    return a;
}

public static double[] func_194c5b13d94e45a98795fcc7de7690c7(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
```

```
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    return len;
}

public static long[] func_69236927ffb84a828938afc50fe059f3(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    return arr;
}

public static long[] func_9340837fe1f84455a091ba6d9ba84d7c(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return a;
```

```
}

public static double[] func_fa7f0094f0ca449c9c2551524bb83604(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    return len;
}

public static long[] func_561b1b0bc59c4c398e8e8e371b6fb942(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return RS;
}

public static int func_04100030570d4e39acc687ae2fa58b4f(int u, int
casenumber, int l, Scanner sc){
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
```

```
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    for (int i = 1; i < l; ++i) {
        areal[i] = areal[i - 1] + (xl[i] - xl[i - 1]) * (yl[i] + yl[i
- 1]);
    // System.out.format("  %d%n", areal[i]);
    }
    for (int i = 1; i < u; ++i) {
        areau[i] = areau[i - 1] + (xu[i] - xu[i - 1]) * (yu[i] + yu[i
- 1]);
    // System.out.format("  %d%n", areau[i]);
    }
    double[] ss = new double[l + u];
    double[] widths = new double[l + u];
    double[] areas = new double[l + u];
    for (int i = 0; i < l + u; ++i) {
        int il = 0;
        int iu = 0;
        while (il < l && xl[il] <= xs[i]) ++il;
        --il;
        while (iu < u && xu[iu] <= xs[i]) ++iu;
        --iu;
        // System.out.format("     %d (%d) %d (%d)%n", il, xl[il], iu,
xu[iu]);
        double a = 0;
        if (xu[iu] == xs[i]) {
            a += areau[iu];
        } else {
            double lambda = (xs[i] - xu[iu]) / ((double) (xu[iu + 1] -
xu[iu]));
            a += areau[iu] + (xs[i] - xu[iu]) * (2 * yu[iu] + lambda *
(yu[iu + 1] - yu[iu]));
        }
        // System.out.format("   %.6f%n", a);
        if (xl[il] == xs[i]) {
            a -= areal[il];
        } else {
            double lambda = (xs[i] - xl[il]) / ((double) (xl[il + 1] -
xl[il]));
            a -= areal[il] + (xs[i] - xl[il]) * (2 * yl[il] + lambda *
(yl[il + 1] - yl[il]));
        }
        areas[i] = a;
        if (xl[il] < xl[l - 1]) {
            ss[i] = (yu[iu + 1] - yu[iu]) / ((double) (xu[iu + 1] -
xu[iu])) - (yl[il + 1] - yl[il]) / ((double) (xl[il + 1] - xl[il]));
            widths[i] = yu[iu] + (yu[iu + 1] - yu[iu]) * (xs[i] -
xu[iu]) / ((double) (xu[iu + 1] - xu[iu]));
```

```
            widths[i] -= yl[il] + (yl[il + 1] - yl[il]) * (xs[i] -
xl[il]) / ((double) (xl[il + 1] - xl[il]));
        }
    }
    System.out.format("Case #%d:%n", casenumber);
    double area = areau[u - 1] - areal[l - 1];
    return g;
}

public static long[] func_d9bd8b85039a43ef94e9cbc8a2bc6f50(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
```

```
    }
    return dp;
}

public static long[] func_b8df90e77e1d45969d91ecaf759bedaf(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    return psum;
}

public static long func_9fa344b14b364dd28017b49085db31e1(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    return sum;
}

public static long[] func_f9afc593f233405d958477da60d38ef4(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
```

```
        ++index;
    }
    double res = 0;
    return dp;
}

public static long[] func_e40ec5b372004be69a4ff904e08fdd14(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    return ar;
}

public static long func_035d1f0d6bab446c9edd8683ccccbce1(int low, int
high, int n, long prefixSum, long[] rsum){
    int mid = (high + low) / 2;
    long sum = (rsum[mid] - prefixSum);
    long suffSum = rsum[n - 1] - sum - prefixSum;
    return sum;
}

public static double func_603f9c70ad8d4cbb80655bfd1fea2bf4(double
chunkLeft, double thisLow, double currentLow, double thisHigh, double
currentHigh){
    currentHigh = thisHigh;
    currentLow = thisLow;
    chunkLeft = 0;
    return currentLow;
}

public static int[] func_012a032ad71f4fc0949a632951d597ef(int n, long
r, long s, long q, long p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) ((((i * p + q) % r) + s));
    }
    return a;
}

public static int func_dcc4537e217d46a5916a8dec258a68f9(Scanner sc){
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    double sl = 0;
    double su = 0;
    return l;
}
```

```java
public static int[] func_4aaa10a0f80c4f6499742a20b867495d(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return arr;
}

public static ArrayList func_1cfe70fed75345cda24f8f342603f280(double
X, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    return lx;
}

public static long func_a51cc61935344f0a8f3df366efa4f12f(int j, int i,
long win, long required, long toAdd){
    required += toAdd * j;
    win += toAdd * i;
    return win;
}

public static long func_60f95200b7704573bd6e9d6de729fa83(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    long best = Long.MAX_VALUE;
    for (int i = 0; i < n; i++) {
        x += a[i];
        Long up = all.ceiling(x / 2 + 1);
```

```
        if (up != null) {
            long now = Math.max(up, Math.max(x - up, sum - x));
            best = Math.min(best, now);
        }
        Long down = all.floor(x / 2);
        if (down != null) {
            long now = Math.max(down, Math.max(x - down, sum - x));
            best = Math.min(best, now);
        }
        all.add(x);
    }
    return best;
}

public static long func_e15961bcbea34f8fb5ac34cd597ae845(int q, int s,
int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    return x;
}

public static long[] func_9eeaf1ab53d841e9bae0c168b6e60c78(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    return arr;
}

public static long[] func_545ffac011e34bb0a9b8fefa99184825(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
```

```
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return best;
}

public static double func_cee8f76b4f9244fc8148812444532b88(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    return d;
}

public static long[] func_b8ca8184c8724186a9f0caf0e7647eb8(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    return A;
}

public static long[] func_7772609e9e79442498cf344c5545e1d0(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
```

```
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    return RS;
}

public static int func_7d8103a79b32433698a2bc462a09d967(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return lo;
}

public static int func_5cee3bc7a7b84b6390e115de246325b4(int c, Scanner
cin){
    int l = cin.nextInt();
    int u = cin.nextInt();
    int g = cin.nextInt();
    int[][] pos = new int[l + u][2];
    for (int i = 0; i < l + u; ++i) {
        pos[i][0] = cin.nextInt();
        pos[i][1] = cin.nextInt();
    }
    System.out.println("Case #" + c + ":");
    return g;
}

public static long[] func_d9da5b3736874fe2b324612e188dd9b9(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
```

```
    }
    return dp;
}

public static int func_df7dc7260e2c43bcad7919d04ec48727(Scanner sc){
    int n = sc.nextInt();
    int p = sc.nextInt();
    int q = sc.nextInt();
    int r = sc.nextInt();
    int s = sc.nextInt();
    long[] a = new long[n];
    return p;
}

public static long func_f19fd8c2f06940898ed765975613008c(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    return sum;
}

public static long func_3c18b615adff48ee8086f7852156c9d5(int
winningThings, long payMoney, long low, long budget, long[] x){
    for (int i = 0; i < winningThings; i++) {
        payMoney += low - x[i];
    }
    if (payMoney > budget) {
        return -1;
    }
    --low;
    return payMoney;
}

public static long[] func_7fae8865ace847e9a9906c2d4bb44d60(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
```

```
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    return pref;
}

public static int func_1d712f0667fa42fb977803240c6c6742(int r, int n,
int p, int q){
    int s = A.readInt();
    A.num = new long[n + 1];
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
```

```
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    A.pw.println(ret);
    return s;
}

public static long func_d486db1d5e8d4a758da285c512fe255b(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return sum;
}

public static long func_3139c2191bdf495face771c096e8400a(int i, int
left, int max, int min){
    int right = (2 * max + min) / 3;
    long leftCount = A.count(i, left);
```

```
    long rightCount = A.count(i, right);
    if (leftCount < rightCount) {
        max = right;
    } else {
        min = left;
    }
    return leftCount;
}

public static double[] func_b4b050734f854b0fb336d13f64aa13cb(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    return len;
}

public static double func_359662e16aa241cba8332c944263268b(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return crt;
}

public static double func_758ff3a375ad4ce4a1eddb1f081af729(double[][]
u, double[][] l){
    double ycurl = l[0][1];
    double ycuru = u[0][1];
    return ycurl;
}

public static long[] func_c4408587338144ffa69186a6488dfcdd(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
```

```
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return arr;
}

public static long[] func_52f438c41325479a967a09b253403a87(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return dp;
}

public static long func_f3726e3da78544fe92db98860fef96ae(int
winningThings, long budget, long lowestBet, long payMoney, long[] x){
    if (payMoney > budget) {
        return -1;
    }
    long ourWin = 0;
```

```
    for (int i = 0; i < winningThings; i++) {
        ourWin += lowestBet - x[i];
    }
    return ourWin;
}

public static long func_7b855ee6312843cba0863cbc1cdf0d47(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return sum;
}

public static double[] func_45be8ec747dc467da824b8e4ef909074(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    return len;
}

public static long func_f36811f17a40400e9ae8d00ca7de72fb(int N, int j,
int i, long sum, long[] dp){
    long a = dp[i + 1];
    long b = dp[j] - dp[i + 1];
    long c = dp[N] - dp[j];
    long left = sum - Math.max(a, Math.max(b, c));
    return c;
}

public static long[] func_250d39f24f5e450fbf0efe210acd5584(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
```

```
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    return LS;
}

public static double func_1040ca9a532f4e1a9f18938a8780e619(A2.Pair p2,
A2.Pair p1){
    int x0 = p1.a;
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    return dy0;
}

public static double func_195a82340e6a421e8c76fbfe4fa3625d(int W, int
i, double g, double firstArea){
    double cutArea = (i + 1) / g * firstArea;
    double min = 0.0;
    double max = W + 0.00000001337;
    return max;
}

public static int[] func_51bdeda202e5479abcb59cef1a00c104(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return arr;
}

public static long func_247865813439470c80070565b0e357af(int N, long
sum, long[] arr, long[] dp){
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
```

```
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return avg;
}

public static long func_db2261c68a5240c99caf787f5b203cff(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    return sum;
}

public static double func_f6fa96cac7364b6c8daa803105f1bca2(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
```

```
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    return res;
}

public static long func_eb5a7301c9b44cd0a1fc6d0e7951a9b6(long sr, long
sm, long sl, long[] a){
    long sum = sm + sr;
    long ans = Math.max(sm, sr);
    int l = 0;
    for (int r = 1; r < a.length; ++r) {
        sm += a[r];
        sr -= a[r];
        while (l < r && Math.max(sl + a[l], sm - a[l]) < Math.max(sl,
sm)) {
            sl += a[l];
            sm -= a[l];
            ++l;
        }
        ans = Math.min(ans, Math.max(sr, Math.max(sm, sl)));
    // System.out.println(" " + l + " " + r + " " + sl + " " + sm + "
" + sr + " " + ans);
    }
    return sum;
}

public static long func_e2fb0f15eb1741659e3112bd0471873d(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
```

```
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return sum;
}

public static double func_b1c326a88b6f4882a861f38d35291ea5(int r, int
s, int n, int p, int q){
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
```

```
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    A.pw.println(ret);
    return ret;
}

public static long func_d77afebe0ad24442a05d0c614fb1aec7(int
winningThings, long budget, long lowestBet, long payMoney, long[] x){
    for (int i = winningThings; i < x.length; i++) {
        if (x[i] <= lowestBet) {
            payMoney += lowestBet + 1 - x[i];
        }
    }
    if (payMoney > budget) {
        return -1;
    }
    return payMoney;
}

public static int func_3a4e681cba6f4dfabe1280989ae13edb(int first, int
N, int second, long avg, long[] arr){
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
```

```
    return index;
}

public static int[] func_4c7eb45097fb40a79a5dde22fdb65c9e(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    return t;
}

public static int func_9583dd0b2a9049989a2280706c59c5c4(int t, Scanner
input){
    System.out.printf("Case #%d: ", t + 1);
    int n = input.nextInt();
    int p = input.nextInt(), q = input.nextInt(), r = input.nextInt(),
s = input.nextInt();
    int[] data = new int[n];
    return r;
}

public static long[] func_ce7aab78e1574819a63838638ad4630e(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return RS;
}

public static int[] func_13849b00c92f44749c3c2ec46af1b0fa(Scanner sc){
    int w = sc.nextInt();
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
```

```
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    for (int i = 1; i < l; ++i) {
        areal[i] = areal[i - 1] + (xl[i] - xl[i - 1]) * (yl[i] + yl[i
- 1]);
    // System.out.format("  %d%n", areal[i]);
    }
    for (int i = 1; i < u; ++i) {
        areau[i] = areau[i - 1] + (xu[i] - xu[i - 1]) * (yu[i] + yu[i
- 1]);
    // System.out.format("  %d%n", areau[i]);
    }
    double[] ss = new double[l + u];
    double[] widths = new double[l + u];
    double[] areas = new double[l + u];
    return xl;
}

public static long func_608cef489bc443daad865a55655333a6(int j, int i,
long required, long[] bets){
    long win = 0;
    for (int k = 0; k < j; k++) {
        required += bets[j - 1] - bets[k];
        if (k < i)
            win += bets[j - 1] - bets[k];
    }
    required += j - i;
    return win;
}

public static long[] func_d3d025f0614340a183428180cb564061(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
```

```
    return A;
}

public static int func_b719d9945af44fc29122933717958341(PrintWriter
out, Scanner in){
    int T = in.nextInt();
    for (int t = 0; t < T; t++) {
        int N = in.nextInt();
        long p = in.nextInt();
        long q = in.nextInt();
        long r = in.nextInt();
        long s = in.nextInt();
        long[] A = new long[N];
        for (int n = 0; n < N; n++) {
            A[n] = (n * p + q) % r + s;
        }
        long[] best = new long[N];
        int beforeHalf = 0;
        long beforeHalfSum = 0;
        long sum = 0;
        for (int n = 0; n < N; n++) {
            sum += A[n];
            while (beforeHalf < n && beforeHalfSum + A[beforeHalf] <
sum / 2) {
                beforeHalfSum += A[beforeHalf];
                beforeHalf++;
            }
            long max = sum;
            if (beforeHalf > 0) {
                max = Math.max(beforeHalfSum, sum - beforeHalfSum);
            }
            if (beforeHalf <= n - 1) {
                long value = Math.max(beforeHalfSum + A[beforeHalf],
sum - beforeHalfSum - A[beforeHalf]);
                max = Math.min(max, value);
            }
            best[n] = max;
        }
        long min = best[N - 1];
        long subSum = 0;
        for (int n = N - 1; n >= 1; n--) {
            subSum += A[n];
            min = Math.min(min, Math.max(subSum, best[n - 1]));
        }
        double answer = (sum - min) / (double) sum;
        out.println("Case #" + (t + 1) + ": " + answer);
    }
    return T;
}
```

```java
public static long[] func_05f511494ded4625b701463b34987d2f(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    return A;
}

public static int func_b806068012884b8abbb99ca9d5f9889e(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return index;
}

public static double func_695eff115f4849a3930ae57db6861fba(A.Point p1,
A.Point p2){
    double dy = p2.y - p1.y;
    double dx = p2.x - p1.x;
    return dy;
}

public static long func_af6a026f475b4b049dd000fc6c298078(int N, int i,
long a, long max, long[] psum){
    long get = psum[N] - Math.max(psum[i], a);
    if (get > max)
        max = get;
    return max;
}

public static double func_6eb7f71cd01f4776ab8e0af619926b4a(int n, int
p, int q, int s, int r){
```

```
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return ans;
}

public static double func_0a511a819d3c479b972cf5035474de00(double a1,
double cx, double c1, double nx, double b1){
```

```
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return mx;
}

public static double func_50dc40dc780b477ebf7ada2012a41281(int r, int
s, int n, int p, int q){
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    A.pw.println(ret);
    return ret;
}

public static double func_d825e1e376df45f0a1651fa4943db848(double y0,
double total, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    for (int i = 0; i < lx.size(); ++i) {
        double xx = lx.get(i);
        double yy = ly.get(i);
        total += (x0 + xx) * (y0 - yy);
        x0 = xx;
        y0 = yy;
    }
    return x0;
}
```

```
public static long func_35b6509aff844c079b305a87539c8829(int n, int r,
int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    // System.out.println(high);
    double ans = 1.0 * (pref[n] - high) / pref[n];
    return high;
}

public static double func_d11649e7f7664034b17f09ff820dc634(int l,
double ans, double right, double left){
    double ax = Math.max(A.lowx[l], left);
    double bx = Math.min(A.lowx[l + 1], right);
    double ay = A.lowy[l];
    double by = A.lowy[l + 1];
    if (ax > A.lowx[l]) {
        ay = A.lowy[l] + (ax - A.lowx[l]) / (A.lowx[l + 1] -
A.lowx[l]) * (A.lowy[l + 1] - A.lowy[l]);
    }
```

```
    if (bx < A.lowx[l + 1]) {
        by = A.lowy[l + 1] + (bx - A.lowx[l + 1]) / (A.lowx[l] -
A.lowx[l + 1]) * (A.lowy[l] - A.lowy[l + 1]);
    }
    ans -= (bx - ax) * ((ay + by) / 2 + 1000);
    return by;
}

public static int[] func_aa7aebfe513d47958d7c3b9bdf0926b6(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    return array;
}

public static long func_ac212b70ff8d43fe9a8439a0b0db407c(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return sum;
}

public static long func_78c9efd0783a42f2a1cb6c2042bf61b1(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
```

```
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalfSum;
}

public static long func_8d08a37f76e042279c85b8a8fa2f3a89(int q, int s,
int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    return x;
}

public static int func_53a5cda45c0d475cb24b25343587fc1a(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
```

```
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return beforeHalf;
}

public static long func_c5e93332c7e44775aa01e4937137e963(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
```

```
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalfSum;
}

public static long func_bf349db88d574a75b893be6d25014872(int
beforeHalf, long max, long sum, long beforeHalfSum, long[] A){
    long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
    max = Math.min(max, value);
    return value;
}

public static long[] func_7751c26a935b4cddb82407e2d551c628(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    return dp;
}

public static int[] func_59f14386d71e4d378c18bbd4ee1d8a42(int r, int
p, int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    return A;
}

public static long func_71a32d5cd6594a08895a5107ea5df1e6(int s, int r,
int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
```

```
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    long total = partials[n];
    return best;
}

public static long[] func_c0a0452676ac483b929e69001e13f4ca(int r, int
p, int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    return partials;
}

public static long[] func_47b547ac22c044c79e52e278ccf71d8e(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
    // sout(i + " " + index);
    }
    return sum;
}
```

```java
public static long func_1ec1871e031648dbbb112dbb1e46094d(int p2, int
p3, int l2, int l3, long l1){
    long eCur = (l1 + l2) * p2 + (l1 + l2 + l3) * (100 - p2) * p3;
    long eSwap = (l1 + l3) * p3 + (l1 + l2 + l3) * (100 - p3) * p2;
    return eCur;
}

public static long[] func_233436063d8e48dabb9fbba19b6f8222(int n, long
r, long p, long q, long s){
    long[] a = new long[n + 1];
    for (int i = 1; i <= n; i++) {
        a[i] = a[i - 1] + ((i - 1) * p + q) % r + s;
    }
    return a;
}

public static long[] func_e7e27a2f805445428105c0f80ac346af(int s, int
N, int q, int p, int r){
    long[] n = new long[N];
    for (int i = 0; i < N; i++) {
        n[i] = ((long) i * p + q) % r + s;
        if (i > 0) {
            n[i] += n[i - 1];
        }
    }
    long ans = Long.MAX_VALUE / 4;
    return n;
}

public static int[] func_2701d1f24855448fbf49eadc6fbdad20(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
```

```
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
        }
    }
    return qty;
}

public static double func_417f4f63b47344f19e82955b9e04f399(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    return b;
}

public static double func_b6a0c8c82f1f40e3b4a3a25ec67e4bec(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    return b;
}

public static double func_08ec6f3e23b24ec5ac209037753fcb7a(int i,
double x2, double bnd, Solution.Point[] p){
    double y2 = p[i + 1].y;
    if (x2 > bnd) {
        double cf = (bnd - p[i].x) / (double) (p[i + 1].x - p[i].x);
        x2 = bnd;
        y2 = p[i].y + (p[i + 1].y - p[i].y) * cf;
    }
    return y2;
}

public static long func_480052bde586495694f96fcf5fed3d7a(int
beforeHalf, long max, long sum, long beforeHalfSum, long[] A){
    long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
    max = Math.min(max, value);
    return value;
}

public static double func_960e22cebc2846ae892bfe21068351a9(double x,
Segment upperSegment){
    double left = upperSegment.a.x;
```

```
    double right = x;
    return left;
}

public static int[] func_d463f2b1cdd74917a668bee7daff106a(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    long total = partials[n];
    return array;
}

public static long[] func_0a19040a5c874b14ab8afac8624c0270(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    while (pow2 * 2 <= N) {
        pow2 *= 2;
    }
    return LS;
}

public static long func_395c0d806e674e289baee44453735f44(int
winningThings, long budget, long lowestBet, long payMoney, long[] x){
```

```
    for (int i = winningThings; i < x.length; i++) {
        if (x[i] <= lowestBet) {
            payMoney += lowestBet + 1 - x[i];
        }
    }
    if (payMoney > budget) {
        return -1;
    }
    long ourWin = 0;
    return payMoney;
}

public static long func_1012635fce1b4f1190db83bc3ecef7af(int rght,
long rs, long sum, long max, long[] pref){
    int left = -1, right = rght;
    while (left < right - 1) {
        int mid = (left + right) >> 1;
        if (pref[mid] * 2 >= sum) {
            right = mid;
        } else {
            left = mid;
        }
    }
    for (int t = right - 2; t <= right + 2; t++) {
        if (0 <= t && t <= rght) {
            long ans = rs;
            ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
            max = Math.min(max, ans);
        }
    }
    return max;
}

public static int func_3902c801a267430d8172d8888c3c8dd4(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
```

```
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return beforeHalf;
}

public static int func_92db1d75c1bb4c18acccd04565fe30d0(int q, int s,
int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
    // sout(i + " " + index);
    }
    return index;
}

public static long func_cd60cb12de624cfda55df29cf04231b3(int n, long
sum, long[] a, TreeSet<Long> all){
    all.add(0L);
    long x = 0;
    long best = Long.MAX_VALUE;
    for (int i = 0; i < n; i++) {
        x += a[i];
        Long up = all.ceiling(x / 2 + 1);
        if (up != null) {
            long now = Math.max(up, Math.max(x - up, sum - x));
            best = Math.min(best, now);
        }
        Long down = all.floor(x / 2);
```

```
        if (down != null) {
            long now = Math.max(down, Math.max(x - down, sum - x));
            best = Math.min(best, now);
        }
        all.add(x);
    }
    return best;
}

public static long func_9de5d49273a04956abd3fb7f816a1b19(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return s2;
}

public static long[] func_78ca30b40e0641959ac900f6ec939c36(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return cum;
}

public static long[] func_61fffa8c747c4b3aaecffd4410c1d7cd(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
```

```
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    return sum;
}

public static long func_3199723b36ce43d7ac9f65f6c2d3d8f8(int n, int p,
int q, int r, Scanner sc){
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    return tmp;
}

public static long[] func_a6c9b2455a4347999552651d27f8286a(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
```

```
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
        }
    }
    return suffixes;
}

public static long func_5cb6cffe27a248219c8695e5ff0deb47(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    for (int i = 0, j = n - 1; i <= j; ) {
        if (si + a[i] < sj + a[j]) {
            si += a[i++];
        } else {
            sj += a[j--];
        }
        min = Math.min(min, Tour.max(sum - si - sj, si, sj));
    }
    return sj;
}

public static int func_f01a916a0fde48a8a78af5b4e414fcac(int lo, int i,
long[] S){
    int hi = i;
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
```

```
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    return ans;
}

public static long func_fea185a186664fbd9212e7381c9e3cc5(long sr, long
sm, long sl, long[] a){
    long ans = Math.max(sm, sr);
    int l = 0;
    for (int r = 1; r < a.length; ++r) {
        sm += a[r];
        sr -= a[r];
        while (l < r && Math.max(sl + a[l], sm - a[l]) < Math.max(sl,
sm)) {
            sl += a[l];
            sm -= a[l];
            ++l;
        }
        ans = Math.min(ans, Math.max(sr, Math.max(sm, sl)));
    // System.out.println(" " + l + " " + r + " " + sl + " " + sm + "
" + sr + " " + ans);
    }
    return sr;
}

public static long[] func_cc56c1becd7f4fd1b4462f268c25461b(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    return a;
}

public static long[] func_95059c6611b74d88ac84ec76d4455e12(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
```

```
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return RS;
}

public static int[] func_debbd2c8d5764e74b67b87d36014f405(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    return a;
}

public static double[] func_2ac03ee314784e9d8c5fb8f541c59f26(int u,
int l, int[] xu, int[] yu, int[] areau){
    for (int i = 1; i < u; ++i) {
        areau[i] = areau[i - 1] + (xu[i] - xu[i - 1]) * (yu[i] + yu[i
- 1]);
    // System.out.format("  %d%n", areau[i]);
    }
    double[] ss = new double[l + u];
    double[] widths = new double[l + u];
    double[] areas = new double[l + u];
    return widths;
}

public static int func_70f4982e179a43f98ab3a8a117da2a1d(int first, int
N, int second, long avg, long[] arr){
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
```

```
        ++index;
    }
    return first;
}

public static int func_db0c9895091741f086d00fbf49ac1f04(int i, int
high, int low, long rest, long[] partials){
    int guess = (low + high) / 2;
    long second = partials[guess] - partials[i];
    if (second * 2 > rest)
        high = guess;
    else
        low = guess;
    return guess;
}

public static int func_72a80e76dd204e20b5f79ab3cd39731b(int n, long
sum, long t1, long[] a){
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    for (int i = 0; i < n; i++) {
        while (j < n && t2 + a[j] <= rest - a[j]) {
            t2 += a[j];
            rest -= a[j];
            j++;
        }
        double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) /
sum;
        ans = Math.max(ans, pp);
        if (j < n) {
            pp = 1.0 * (sum - Math.max(t1, Math.max(t2 + a[j], rest -
a[j]))) / sum;
            ans = Math.max(ans, pp);
        }
        t1 += a[i];
        t2 -= a[i];
        if (j == i) {
            rest -= a[j];
            j++;
        }
    }
    return j;
}

public static long func_51260ba8b9664efea6361c62a03b60e5(int lo, int
i, int n, long max, long[] S){
    int hi = i;
    int ans = -1;
```

```
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return tot;
}

public static int func_ae50b5969982447cafdb5c9a3d9ebcda(Scanner cin){
    int w = cin.nextInt();
    int l = cin.nextInt();
    return w;
}

public static double func_1e3b4737809743db8a8c80127b593983(int n, int
G, Integer[] xxx, int[][] y, int[][] x){
    double[][] y1 = new double[2][n];
    int a = 0, b = 0;
    for (int i = 0; i < n; i++) {
        while (xxx[i] > x[0][a + 1]) a++;
        while (xxx[i] > x[1][b + 1]) b++;
        y1[0][i] = (y[0][a + 1] - y[0][a]) * 1.0 * (xxx[i] - x[0]
[a]) / (x[0][a + 1] - x[0][a]) + y[0][a];
        y1[1][i] = (y[1][b + 1] - y[1][b]) * 1.0 * (xxx[i] - x[1]
[b]) / (x[1][b + 1] - x[1][b]) + y[1][b];
    }
    double[] ans = new double[G - 1];
    double S = 0.0;
    for (int i = 0; i + 1 < n; i++) {
```

```
        double dy1 = y1[1][i] - y1[0][i], dy2 = y1[1][i + 1] - y1[0][i
+ 1], dx = xxx[i + 1] - xxx[i];
        S += (dy1 + dy2) * dx / 2.0;
    }
    double S_left = 0.0;
    int j = 1;
    return S;
}

public static long[] func_060c2681775a4bd4b9a34d0f1bd1da03(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return A;
}

public static int[] func_307d10b7dfcb4809ba38b0ed3a25e860(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    return a;
}

public static int func_7c7bd9e2075644e5afac738548c569a4(int i, long
initadd, long[] v){
    long waste = 0;
    int numtie = 0;
```

```
    for (int j = i; j < 37; j++) {
        if (v[j] == v[i]) {
            numtie++;
        }
    }
    if (v[i] == v[i - 1]) {
        v[i]++;
        initadd++;
        waste += numtie;
    }
    long bet = i * (v[i] - 1);
    return numtie;
}

public static double[] func_22857e768be049a5bdc81cf6a4c1ab78(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
```

```
        len[i] = upper[i] - low[i];
    }
    return len;
}

public static long func_a613b6e4b4a54056835fb994fd5fb775(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return beforeHalfSum;
}

public static int[] func_4002286777fc4056b65113ca1ee9a5a4(int r, int
p, int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return A;
}

public static long func_0babe772a16e4c098d63aeff28d54e99(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return avg;
}

public static long func_82e52f9899124a98b24f71fca683028b(int n,
Scanner in){
```

```
    long P = in.nextInt();
    long Q = in.nextInt();
    long R = in.nextInt();
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    return R;
}

public static double func_ca1d8633e521436793ab2f2b87c40cd7(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    lastx = mid;
    return hi;
}

public static double func_f97c0451914a460d82887d1fbd2731fc(double X,
double upY, double[] angleDown, double[] down, double[] up){
    double downY = (X - (int) X) * angleDown[(int) X + 1] + down[(int)
X];
    double St = (upY - downY + up[(int) X + 1] - down[(int) X + 1]) *
((int) X + 1 - X) / 2;
    return St;
}

public static double[] func_65401df27b4d40b7a04b122bf84b7c12(int w,
int[] lx, int[] ly, int[] ux, int[] uy){
    double[] h = new double[w + 1];
    for (int i = 0, pu = 0, pl = 0; i <= w; i++) {
        double x = i;
        if (i > ux[pu + 1])
            pu++;
        if (i > lx[pl + 1])
            pl++;
        double lH = (ly[pl] * (lx[pl + 1] - x) + ly[pl + 1] * (x -
lx[pl])) / (lx[pl + 1] - lx[pl]);
        double uH = (uy[pu] * (ux[pu + 1] - x) + uy[pu + 1] * (x -
```

```
ux[pu])) / (ux[pu + 1] - ux[pu]);
        h[i] = uH - lH;
    }
    double[] S = new double[w];
    double tS = 0;
    return h;
}

public static double func_69c4e5fffd26468aa88725016fe2635c(double
guess, double targetArea, double low, double curans, double high){
    double a = A.area(curans, guess);
    if (a < targetArea) {
        low = guess;
    } else {
        high = guess;
    }
    return low;
}

public static double func_f4d444d2e7aa424db228e0f2f819f9e2(double
chunkLeft, double thisLow, double currentLow, double thisHigh, double
currentHigh){
    currentHigh = thisHigh;
    currentLow = thisLow;
    chunkLeft = 0;
    return currentLow;
}

public static long func_1c0963c92fb0498db75e479b608600b9(int n, int r,
long cur, long best, long[] s){
    cur = Math.max(cur, s[n] - s[r]);
    best = Math.min(best, cur);
    return best;
}

public static long[] func_6ba5775e23e5499291e52b8d0f265f73(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    return psum;
}

public static long func_2022dd820dfb48c9935edae8a6e05ef3(int N, long
p, long r, long s, long q){
```

```
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return sum;
}

public static int func_e0cb80603cfb44e1b328acc70cae7888(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return hi;
}

public static long func_a5e389d27400492d9a4fbeefcc20670a(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    long max = 0;
    for (int i = 0; i < N; i++) {
        int lo = i + 1, hi = N;
        long left = psum[N] - psum[i];
        while (lo < hi) {
            int mid = (lo + hi) >> 1;
            long s1 = psum[mid] - psum[i];
            long s2 = left - s1;
            if (s1 >= s2) {
                hi = mid;
            } else {
                lo = mid + 1;
            }
```

```
        }
        long t = psum[lo] - psum[i];
        long a = Math.max(t, left - t);
        if (lo > i + 1) {
            t = psum[lo - 1] - psum[i];
            a = Math.min(a, Math.max(t, left - t));
        }
        long get = psum[N] - Math.max(psum[i], a);
        if (get > max)
            max = get;
    }
    magictour.out.printf("%.15f\n", (double) max / (double) psum[N]);
    return max;
}

public static long func_78965f513b784baeb74537b427cf35b5(int lo, int
i, int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    return take;
}

public static int[] func_4c628980499b47ef87e463e377bd8170(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
```

```
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    return a;
}

public static int func_5ebcc7c464bf43e19153edbda1a0d7d4(int i, int
left, int max, int min){
    int right = (2 * max + min) / 3;
    long leftCount = A.count(i, left);
    return right;
}

public static double func_26aa31e215434fa995afd00b711e36fa(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    // System.out.println(high);
    double ans = 1.0 * (pref[n] - high) / pref[n];
    return ans;
```

```
}

public static int func_7c247de502ac4e41bb279832580c762d(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalf;
}

public static long[] func_63732a4edf6d4370af71d5fee05bcb3f(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    return a;
}
```

```
public static double func_ea3e7600d3034570ae8800782ede5f8e(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return res;
}

public static int func_67508bfe8e2c4975ab8e187fb9bcc6fe(int p, int q,
int n, Scanner scanner){
    int r = scanner.nextInt();
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    return s;
}

public static long func_6c34dc45560c43a4828be261104e5da9(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    for (int i = 0, j = n - 1; i <= j; ) {
        if (si + a[i] < sj + a[j]) {
```

```
            si += a[i++];
        } else {
            sj += a[j--];
        }
        min = Math.min(min, Tour.max(sum - si - sj, si, sj));
    }
    return sj;
}

public static ArrayList func_c5056cb41c774f588cd2cb5677c04735(double
X, double y0, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    return lx;
}

public static double func_420ad025277b47369679990ab04a5336(int W, int
U, int G, int L, Scanner input){
    CodeJam_Round3_A.lower = new double[L][2];
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
```

```
    }
    return firstArea;
}

public static long func_de895c965086445b853ab9e83f0885f5(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    for (int i = 2, j = 1; i < n; i++) {
        while (j + 1 < i && TaskA.sum(0, j + 1) <= TaskA.sum(j + 1,
i)) {
            ++j;
        }
        minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i, n),
TaskA.sum(j, i)), TaskA.sum(0, j)));
        if (j + 1 < i) {
            minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i,
n), TaskA.sum(j + 1, i)), TaskA.sum(0, j + 1)));
        }
    }
    return minimal;
}

public static int func_bf7124a6f322474a913d87d75e2f6f4b(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
```

```
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    return second;
}

public static double[] func_a404004db9a045828595caa0b9058b06(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
```

```
        area += (len[i - 1] + len[i]) / 2;
    }
    return low;
}

public static long func_41d608421fb94615bd45dc5e91188ad6(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return beforeHalfSum;
}

public static int[] func_d4cd59e6a5bc4545ac5fd028212e9a90(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    return arr;
}

public static long[] func_93e6fe1efa494b309d4362429ea67c59(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    return rsum;
}
```

```java
public static long func_534e9a17183847e4962d477001d4e637(int n, int r,
int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    return high;
}

public static int func_ab40cbd7cdf54cd88df1bfceb0575b00(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
```

```
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return index;
}

public static long func_df44cfb881b045cebc4be23f36061513(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
```

```
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return r;
}

public static long func_225ce11136f546b5a38eee73e7b36ada(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return tot;
}

public static double func_a4fa025d1aac4830a34e774393d42bad(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
```

```
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return res;
}

public static long[] func_824efb8788624481bdead75016d0d510(int r, int
s, int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    return a;
}

public static long[] func_a439bc0079144299bf6cc06702c525ed(int n, long
p, long q, Scanner br){
    long r = br.nextInt();
    long s = br.nextInt();
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    return rsum;
}

public static long[] func_51eb8c04b67c47adaf61c71d225c5f5b(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
```

```
    long[] pref = new long[n];
    for (int i = 0; i < n; i++) {
        pref[i] = ar[i];
        if (i > 0) {
            pref[i] += pref[i - 1];
        }
    }
    long max = Long.MAX_VALUE;
    for (int rght = 0; rght < n; rght++) {
        long rs = pref[n - 1] - pref[rght];
        long sum = pref[rght];
        int left = -1, right = rght;
        while (left < right - 1) {
            int mid = (left + right) >> 1;
            if (pref[mid] * 2 >= sum) {
                right = mid;
            } else {
                left = mid;
            }
        }
        for (int t = right - 2; t <= right + 2; t++) {
            if (0 <= t && t <= rght) {
                long ans = rs;
                ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
                max = Math.min(max, ans);
            }
        }
    }
    return ar;
}

public static int[] func_9d8e0bbf52304ab284573e2443c8bdf7(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    return ps;
}

public static int func_07a239d29b5943a188a1cbbc53fab8f3(int N, long
sum, long avg, long[] arr, long[] dp){
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
```

```
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return second;
}

public static int func_47918277cc02444c908bc9bf20caf970(int W, int L,
int t, Scanner input){
    int U = input.nextInt();
    int G = input.nextInt();
    CodeJam_Round3_A.lower = new double[L][2];
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
```

```
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    System.out.println("Case #" + t + ":");
    for (int i = 0; i < cuts.length; i++) System.out.println(cuts[i]);
    return G;
}

public static double func_669bb2d8eb494a50a0a2e1ce8889fb7d(int i,
double S, double low, double high, PrintWriter pw){
    for (int kk = 0; kk < 77; ++kk) {
        double mid = (low + high) / 2;
        double SS = (BaiA.calc(mid, BaiA.U, BaiA.xu, BaiA.yu) -
BaiA.calc(mid, BaiA.L, BaiA.xl, BaiA.yl));
        if (SS < S * (i + 1))
            low = mid;
        else
            high = mid;
    }
    pw.printf("%.8f\n", low);
    return high;
}

public static int func_1d7822e003754975811c091dfef74ee9(int i, int
high, int low, long rest, long[] partials){
    while (low + 1 < high) {
        int guess = (low + high) / 2;
        long second = partials[guess] - partials[i];
        if (second * 2 > rest)
            high = guess;
        else
            low = guess;
    }
    return low;
}

public static Point func_592e5c6dee6643bab02707ed88d77678(double x,
Segment upperSegment){
    Line vertical = new Point(x, 0).line(new Point(x, 1));
    Point upperIntersection = upperSegment.line().intersect(vertical);
    return upperIntersection;
```

```
}

public static long[] func_c578a3022c47400887fc3bece371868b(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    double res = (A[N] - ans) * 1.0 / A[N];
    return A;
}

public static long func_d3ff9646dede42c4a9b9b4b72374edac(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    for (int i = 2, j = 1; i < n; i++) {
        while (j + 1 < i && TaskA.sum(0, j + 1) <= TaskA.sum(j + 1,
i)) {
            ++j;
        }
        minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i, n),
TaskA.sum(j, i)), TaskA.sum(0, j)));
        if (j + 1 < i) {
            minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i,
```

```
n), TaskA.sum(j + 1, i)), TaskA.sum(0, j + 1)));
        }
    }
    return minimal;
}

public static double func_e63397dfd85547c7bbd12d639030cf55(int i,
int[] xl, int[] yl, double sm, double m){
    double xp = xl[i];
    double yp = yl[i];
    if (xl[i] >= m) {
        xp = m;
        yp = yl[i - 1] + (yl[i] - yl[i - 1]) * (xp - xl[i - 1]) / (0.0
+ xl[i] - xl[i - 1]);
    }
    sm -= (xp - xl[i - 1]) * (yp + yl[i - 1]) / 2.0;
    return sm;
}

public static long func_01aed04e7f644a729c54576b5800d43e(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return lo;
}

public static int[] func_df548635fa3046aa8b5f3474ace93215(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    return qty;
}
```

```
public static long[] func_5c91b2b586e547c5bd0d7beb3ecad8d2(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return rsum;
}

public static int func_0c75b9887e3249c5af7bf76870ac53b9(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
```

```
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return second;
}

public static long func_b49e1eb95cba496d8cec22cdf548be6e(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
```

```
            } else {
                l = m;
            }
        }
    }
    return r;
}

public static long func_949d61175bff45f88250aec23c34ee21(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    return t1;
}

public static long func_528a7f9849b84a6c907f79b41233dbdc(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return t1;
}

public static double func_0c3c55a1fc974088abdf56ca66f60390(int n, int
p, int q, Scanner sc){
    int r = sc.nextInt();
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
```

```
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return ans;
}

public static long[] func_11eb94d4304941a4bfc976f17c50b1d7(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    final long total = sums[N];
    long answer = 0L;
    return sums;
}
```

```java
public static int func_41af27006559437488db76f22465de4e(int n, int p,
Scanner sc){
    int q = sc.nextInt();
    int r = sc.nextInt();
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    return q;
}

public static int func_7a3e07108b9a4d788081e7ac20ae326d(int lo, int N,
int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    if (lo > i + 1) {
        t = psum[lo - 1] - psum[i];
        a = Math.min(a, Math.max(t, left - t));
    }
    long get = psum[N] - Math.max(psum[i], a);
    return hi;
}

public static long func_f6b049a70c4340318fbd157a872b371a(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
```

```
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalfSum;
}

public static long func_9d93c5f4803e4471baaca57a7dfd02c4(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return R;
}

public static long func_aa0fc80067a74739917cb0ed835d98c4(int lo, int
N, int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
```

```
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    if (lo > i + 1) {
        t = psum[lo - 1] - psum[i];
        a = Math.min(a, Math.max(t, left - t));
    }
    return t;
}

public static long func_e0822341df994c0f869cf192caf83e70(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return sum;
}

public static int[] func_6bc1f73e8aa24de883192c24b86c193c(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    return array;
}

public static long func_43870017d65b4b52bb15ac5ee9b10b4a(int n, int p,
int q, int r, int s){
```

```
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    return lo;
}

public static double func_9f2a9f078f244913880f59a48baeabf2(double x,
Segment upperSegment){
    double left = upperSegment.a.x;
    double right = x;
    return right;
}

public static double func_451e30aaa4de47498322b2983c33c852(double m,
double[] angleDown, double[] angleUp, double[] down, double[] up){
    double upY1 = (m - (int) m) * angleUp[(int) m + 1] + up[(int) m];
    double downY1 = (m - (int) m) * angleDown[(int) m + 1] +
down[(int) m];
    return upY1;
}

public static long func_c04d2b62302f4ae0bc94ba521c40a310(int lo, int
i, long[] S){
    int hi = i;
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
```

```
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    return diff2;
}

public static double func_4be973b975c74936a6211246bc3cf763(A.Point[]
low, A.Point[] high){
    int[] events = A.getEvents(low, high);
    double area = 0;
    return area;
}

public static int func_ebc1dc1a5a5c4465a23bb57c42dad8a6(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    int Tba = 100 * Lb + La * (100 - Fb);
    return Fb;
}

public static double func_e731cd747e70471580317a1fcfce210a(int n, int
x, int y, long sum, long[] a){
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
```

```
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return ans;
}

public static long func_c072fca00f7a4f368e5fc0cd9a97641c(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return leftSum;
}

public static int func_b1580430e99e4f64a205a3ff9da5a516(int first, int
N, int second, long avg, long[] arr){
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
```

```
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return index;
}

public static long func_3a8d061212a14320b49b6edcaf1cabe4(int n, long
sum1, long mid, long[] a){
    int first = 0;
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
        second--;
    }
    return sum2;
}

public static double func_d60e9af7e54949628a0c9a92ab54ca31(int minCol,
long need2, long level, long budget, long need1){
    if (need1 + need2 > budget) {
        return -1;
    }
    double profit = level * 36.0 * need1 / (level * minCol) - (need1 +
need2);
    return profit;
}

public static int func_c7c635410c224a44880c8c437b036168(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    return med2;
}

public static long func_cf92c0352d0542299ac423261bfa5cb5(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
```

```
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return beforeHalfSum;
}

public static long[] func_1fc5410f27a1493f9b6930656b42a2d3(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return LS;
}

public static int func_42693c5c0b0b4e04b20110a3bf983c56(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
```

```
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return index;
}

public static long[] func_c004c44cb0c3460b9b310d9784bfbe9c(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    return a;
}

public static long func_463c1504e7e540c4aa448102b4a3e8de(int n, int b,
int a, int i, long[] sum){
    long s1 = sum[b - i + 1] - sum[a];
    long s2 = sum[a];
    long s3 = sum[n] - sum[b - i + 1];
    return s3;
}

public static long func_f1cf61481f024fa5b65ab627729fd591(int n, long
prefixSum, long sum, long best, long[] rsum){
    long suffSum = rsum[n - 1] - sum - prefixSum;
    long max = Math.max(sum, suffSum);
    max = Math.max(max, prefixSum);
    best = Math.max(best, rsum[n - 1] - max);
    return best;
}

public static double func_0563cb3eec7b46a5bbb323374295ac07(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
```

```
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return res;
}

public static int func_c63f7f2e969441769f28c775e95eefc7(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
```

```
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return index;
}

public static int[] func_66ad35d3e9e541769ee6dc8aedd6012d(Scanner in){
    int w = in.nextInt();
    int l = in.nextInt();
    int u = in.nextInt();
    int g = in.nextInt();
    int[] x1 = new int[l], y1 = new int[l], x2 = new int[u], y2 = new
int[u];
    for (int i = 0; i < l; i++) {
        x1[i] = in.nextInt();
        y1[i] = in.nextInt();
    }
    return x2;
}

public static double func_3b50df8a644b4c37abf8d37c6d8722b6(int llidx,
double d2, double d3, double xe, double d1){
    d3 = xe - A.l[llidx].x;
    double ysl = A.l[llidx].y + (d1 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    return ysl;
}

public static double func_10bbeddf21d349eba0f789b8ab578485(double yy,
double xx, double y0, double total, double x0){
    total += (x0 + xx) * (y0 - yy);
    x0 = xx;
    return x0;
}

public static long[] func_7a5130a866144805b4d79a022ec3ec48(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
```

```
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return RS;
}

public static long[] func_fe75f9639b1e41efbe9e07373278252c(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    return a;
}

public static double func_02d20e9ada3a4bbf9f767e511607fdf7(int l,
double ans, double ax, double ay, double bx){
    double by = A.lowy[l + 1];
    if (ax > A.lowx[l]) {
        ay = A.lowy[l] + (ax - A.lowx[l]) / (A.lowx[l + 1] -
A.lowx[l]) * (A.lowy[l + 1] - A.lowy[l]);
    }
    if (bx < A.lowx[l + 1]) {
        by = A.lowy[l + 1] + (bx - A.lowx[l + 1]) / (A.lowx[l] -
A.lowx[l + 1]) * (A.lowy[l] - A.lowy[l + 1]);
    }
    ans -= (bx - ax) * ((ay + by) / 2 + 1000);
    return ans;
}

public static long func_3c12ca24c76c4b0aa9a46e539f4a217b(int lo, int
i, int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
```

```
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    return left;
}

public static long func_496ed421057c4edb94df8490f579922d(int start,
int N, long[] sums){
    int left = start;
    int right = N;
    while (left <= right) {
        final int middle = (left + right) >> 1;
        final long midCount = sums[middle] - sums[start];
        final long rightCount = sums[N] - sums[middle];
        if (midCount < rightCount) {
            left = middle + 1;
        } else {
            right = middle - 1;
        }
    }
    long minSolveig = Long.MAX_VALUE;
    return minSolveig;
}

public static long func_43dc08cfa31241beab5c58d56b368f2b(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    return sum;
}
```

```java
public static long[] func_10d450e428b7474a854f84e6e3e811db(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    return partial;
}

public static int func_48a51edc8e8f43bd9c141971f348406f(int r, int p,
int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return s;
}

public static long func_0346aea4dcfe4e208c5ce4ffdeed1370(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return sum;
}

public static long[] func_36a5faba0dac444b80e1478c97a7e6ad(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
```

```
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    // System.out.println(high);
    double ans = 1.0 * (pref[n] - high) / pref[n];
    return pref;
}

public static long func_90792ac4ed904713abae03283950956f(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
```

```
    temp = 0;
    return temp;
}

public static long[] func_c0d604960dc949689d852d83ee83b92c(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return arr;
}

public static int func_e427842888ae4cb1b5da4b0e0f1171e4(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return first;
}

public static long func_103c0892180047e08397f9da991d1e01(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
```

```
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return L;
}

public static int[] func_dbe81bd026dc4b12941a438e6d61b2b7(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
    // sout(i + " " + index);
    }
    return a;
}

public static long[] func_775fbc8b765e49ae8d391af8f28dff19(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
```

```
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    Locale.setDefault(Locale.US);
    return sum;
}

public static long func_d70dedc56b2d4ae9aca4cbb409d9e10e(int r, int p,
int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return max;
}

public static long[] func_cb92124325234ebbbde1ede61396cf32(int N, long
q, long s, long r, long p){
```

```
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    return a;
}

public static long func_0837e70ea50d49eea8ddd1a502fc4131(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return rest;
}

public static long func_3d5461ba2bc64085b58cb2a884ba7552(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
```

```
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
            }
        }
        return answer;
    }

    public static long func_cd5697535dfe43acafedccb052cfc6ae(int
winningThings, long payMoney, long low, long budget, long[] x){
        for (int i = 0; i < winningThings; i++) {
            payMoney += low - x[i];
        }
        if (payMoney > budget) {
            return -1;
        }
        --low;
        long high = budget + 1;
        while (high - low > 1) {
            long mid = low + high >> 1;
            if (A.canDo(budget, x, winningThings, mid)) {
                low = mid;
            } else {
                high = mid;
            }
        }
        return low;
    }

    public static long func_3d8594eba2454326a4cf4e2f1448e4bb(int N, long
q, long p, long r, long s){
        long[] A = new long[N];
        for (int n = 0; n < N; n++) {
            A[n] = (n * p + q) % r + s;
        }
        long[] best = new long[N];
        int beforeHalf = 0;
        long beforeHalfSum = 0;
        long sum = 0;
        for (int n = 0; n < N; n++) {
            sum += A[n];
            while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
                beforeHalfSum += A[beforeHalf];
                beforeHalf++;
            }
            long max = sum;
            if (beforeHalf > 0) {
                max = Math.max(beforeHalfSum, sum - beforeHalfSum);
            }
            if (beforeHalf <= n - 1) {
```

```
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return beforeHalfSum;
}

public static double func_05a4814973e64968ae54fe99e0c57a2d(double
thisLow, double currentLow, double thisHigh, double currentHigh){
    currentHigh = thisHigh;
    currentLow = thisLow;
    return currentLow;
}

public static long[] func_6bc4806fab1f42ef87d9995582c7a7d4(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
```

```
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return best;
}

public static long func_168474d8931c4791a6ccaaab247bb650(long c, long
b, long a){
    long s = Math.max(a, b);
    s = Math.max(s, c);
    return s;
}

public static long func_d4e7a5bb29f44331b1eb5693a6e5c8f0(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    return ans;
}

public static int[] func_ab7e5275e0ae44fe8470dbbadf423043(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
    // sout(i + " " + index);
    }
    return a;
}
```

```java
public static int func_4e639f1c8a5741548633a4b1d3c4f2f5(int l, int g,
int u, int w, int[][] pos){
    double[] bound = new double[w + 1];
    int px = pos[0][0];
    int py = pos[0][1];
    for (int i = 1; i < l; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] -= (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    px = pos[l][0];
    py = pos[l][1];
    for (int i = l + 1; i < l + u; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == l + 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] += (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    double sum = 0.0;
    for (int i = 0; i < w; ++i) {
        sum += (bound[i] + bound[i + 1]) / 2.0;
    }
    double req = sum / g;
    double cur = 0.0;
    return py;
}

public static long func_5655b63df2b942d28335e47f9dcf0b91(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return sum;
}

public static double func_b5e37646b25b4d9db5df5c900b2490ba(int N, long
```

```
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return ans;
}

public static long[] func_03b625f0230b461f86970086bb140010(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
```

```
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return best;
}

public static double func_0b73da7b2c344a38b68a5382d8a27d5a(int G,
Integer[] xxx, int[][] y, int[][] x){
    int n = xxx.length;
    double[][] y1 = new double[2][n];
    int a = 0, b = 0;
    for (int i = 0; i < n; i++) {
        while (xxx[i] > x[0][a + 1]) a++;
        while (xxx[i] > x[1][b + 1]) b++;
        y1[0][i] = (y[0][a + 1] - y[0][a]) * 1.0 * (xxx[i] - x[0]
[a]) / (x[0][a + 1] - x[0][a]) + y[0][a];
        y1[1][i] = (y[1][b + 1] - y[1][b]) * 1.0 * (xxx[i] - x[1]
[b]) / (x[1][b + 1] - x[1][b]) + y[1][b];
    }
    double[] ans = new double[G - 1];
    double S = 0.0;
    for (int i = 0; i + 1 < n; i++) {
        double dy1 = y1[1][i] - y1[0][i], dy2 = y1[1][i + 1] - y1[0][i
+ 1], dx = xxx[i + 1] - xxx[i];
        S += (dy1 + dy2) * dx / 2.0;
    }
    double S_left = 0.0;
    int j = 1;
    return S_left;
}

public static double func_f205599e6cc045a99486bb0785151f81(double X,
double y0, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    return y0;
}

public static int[] func_de58e8e41e5a4b26b5aeca9bc63fc506(int n, int
r, int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
```

```
        sum += values[i];
    }
    return values;
}

public static long[] func_3a13d48fc0574a62b35b1396a9ecb832(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    return sum;
}

public static int func_5b27e23d506547c98a073cc2218004ec(Scanner sc){
    int w = sc.nextInt();
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    for (int i = 0; i < l; ++i) xs[i] = xl[i];
    for (int i = 0; i < u; ++i) xs[l + i] = xu[i];
    Arrays.sort(xs);
    int[] areal = new int[l];
    int[] areau = new int[u];
    for (int i = 1; i < l; ++i) {
        areal[i] = areal[i - 1] + (xl[i] - xl[i - 1]) * (yl[i] + yl[i
- 1]);
    // System.out.format("  %d%n", areal[i]);
    }
    return g;
}

public static long[] func_839bd500038f48008c00470c9f77d810(int N, long
r, long s, long p, long q){
```

```java
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    final long total = sums[N];
    long answer = 0L;
    return sums;
}

public static int func_736b8b7a278c4ce383afb1630274ff0d(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    return x0;
}

public static int[] func_bf6aa248e1694fcea9fe6450ba4a45db(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    return ps;
}

public static long func_9a975dec95354694b816cc5045788529(int middle,
int start, int N, long[] sums){
    final long midCount = sums[middle] - sums[start];
    final long rightCount = sums[N] - sums[middle];
    return rightCount;
}

public static double func_8d0a6976caeb449fb46cde285992fa8e(double
ynewl, double xnew, double ycurl, double xcur){
    xcur = xnew;
    ycurl = ynewl;
    return ycurl;
}
```

```
public static long func_61d576d08b584a85aadc9bef281e761e(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    for (int i = 0; i < n; i++) {
        while (j < n && t2 + a[j] <= rest - a[j]) {
            t2 += a[j];
            rest -= a[j];
            j++;
        }
        double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) /
sum;
        ans = Math.max(ans, pp);
        if (j < n) {
            pp = 1.0 * (sum - Math.max(t1, Math.max(t2 + a[j], rest -
a[j]))) / sum;
            ans = Math.max(ans, pp);
        }
        t1 += a[i];
        t2 -= a[i];
        if (j == i) {
            rest -= a[j];
            j++;
        }
    }
    return t1;
}

public static long func_5c9bb67f398c4a1391a215a7737668c6(int i, int
ans, int n, long max, long[] S){
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
```

```
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return center;
}

public static long func_e72e1cc9a69447e8a42e22ace130a180(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return subSum;
}

public static long[] func_13ab288dc3264b37b3d6e9f3f1312c0a(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
```

```
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return A;
}

public static long func_570f77f5cf5a4fa396d5f0c685c9741d(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
```

```
    }
    return L;
}

public static int[] func_63e35001b4054effb396f06e95b71af0(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    return a;
}

public static int func_31271070b5944c9db6eab6be2e80b902(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return lo;
}

public static long func_29fc833a507c4f8c979248214abede8c(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
```

```
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return r;
}

public static long func_00aed984f3ac49768b06aeeebfc0c067(int n, int s,
int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
```

```
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    return left;
}

public static long[] func_ae4a46b1fb48407881e3eeac9cfde8db(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    return a;
}

public static double func_bd2dc02e8a774284810d122a35f95890(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
```

```
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return answer;
}

public static double func_c8fd6a92ff9a461c972338a8a79f5566(int ulidx,
int llidx, double x1, double x2){
    // System.out.println(x1 + " " + ulidx + " " + llidx);
    double xe = Math.min(Math.min(A.u[ulidx + 1].x, A.l[llidx + 1].x),
x2);
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    return xe;
}

public static int func_4b35e96e8f3441859b54d2f59eb2af87(int lo, int i,
int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
```

```
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    return hi;
}

public static long func_eadb53df7169434a9ba4d1d168da2e9d(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    return sum;
}

public static int[] func_f07417b7ce564efe882715df83f0da0c(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return arr;
}

public static double[] func_134a1cd917674e69a51427fe853fec68(int t,
int w, Scanner in){
    int l = in.nextInt();
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
```

```
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    return len;
}

public static long func_ba9ee46653844115bd8381a73a774a66(int r, int p,
int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return max;
}

public static double func_3ba6d3f50ba4447ca45fef36f8ecd69e(double
guess, double targetArea, double low, double curans, double high){
    double a = A.area(curans, guess);
    if (a < targetArea) {
        low = guess;
    } else {
        high = guess;
    }
    return a;
}

public static double func_e6e041b9cd2946638eca83db4a573438(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return cy1;
}

public static long[] func_ac8c3bd70f6544a59306b3211fa3efdf(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
```

```
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    return rsum;
}

public static double[] func_83f90e00514748e8aba00c8e04e75202(int w,
int[] lx, int[] ly, int[] ux, int[] uy){
    double[] h = new double[w + 1];
    for (int i = 0, pu = 0, pl = 0; i <= w; i++) {
        double x = i;
        if (i > ux[pu + 1])
            pu++;
        if (i > lx[pl + 1])
            pl++;
        double lH = (ly[pl] * (lx[pl + 1] - x) + ly[pl + 1] * (x -
lx[pl])) / (lx[pl + 1] - lx[pl]);
        double uH = (uy[pu] * (ux[pu + 1] - x) + uy[pu + 1] * (x -
ux[pu])) / (ux[pu + 1] - ux[pu]);
        h[i] = uH - lH;
    }
    double[] S = new double[w];
    return h;
}

public static long func_c8aedc20b38c4f58bd8e2e21d48df81d(int j, int N,
int i, long ans, long[] A){
    long third = A[N] - A[i + 1];
    while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
    {
        long first = A[j + 1];
        long second = A[i + 1] - A[j + 1];
        ans = Math.min(ans, Math.max(first, Math.max(second, third)));
    }
    if (j < i) {
        long first = A[j + 2];
        long second = A[i + 1] - A[j + 2];
        ans = Math.min(ans, Math.max(first, Math.max(second, third)));
    }
    return ans;
}

public static double func_d0aa06a2b44e4a7b8bef7e0b978f47ec(double t,
double k, double d1, double want, double dx){
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
```

```
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
    }
    return temp;
}

public static long[] func_40d75750fce748e680801b9ce6caf128(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    // System.out.println(high);
    double ans = 1.0 * (pref[n] - high) / pref[n];
    return pref;
}

public static int[] func_5fa3877939a643d6a293bd639f1d30cf(int r, int
```

```
p, int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    for (int i = 0; i < n; i++) {
        int lo = 0;
        int hi = i;
        int ans = -1;
        long tot = S[i + 1] - S[0];
        while (lo <= hi) {
            int mid = (lo + hi) / 2;
            long center = S[i + 1] - S[mid];
            long left = tot - center;
            // System.out.println(mid + " " + left + " " + center);
            if (left > center) {
                hi = mid - 1;
            } else {
                ans = mid;
                lo = mid + 1;
            }
        }
        // System.out.println(i + " " + ans);
        long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
        long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
        if (diff2 < diff1)
            ans++;
        // System.out.println(i + " " + ans + " " + diff1 + " " +
        // diff2);
        long center = S[i + 1] - S[ans];
        long left = S[ans] - S[0];
        long right = S[n] - left - center;
        long take = S[n] - Math.max(center, Math.max(left, right));
        max = Math.max(max, take);
    }
    return A;
}

public static long func_b8f35a14cf094f9cbed28f548aeeb52a(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
```

```
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    return sum;
}

public static long[] func_f8cb451890eb453bbcc4a99a48c53668(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    return cum;
}

public static long func_2912fbc8cfcd4e1cab9731b4b4eaa94b(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
```

```
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return r;
}

public static double func_f5f3bb09845e4c3d9e7732e6432b5901(double t,
double k, double d1, double dx){
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    return temp;
}

public static double func_ebcb5f8d7c414a7e9af8bc4f0a4377b0(int t, int
w, int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
```

```
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
    return area;
}

public static double func_fee3ce9fabd64b7281da67fe9f9cc193(double
target, double hi, double mid, double lastx, double lo){
    mid = (hi + lo) / 2;
    double ar = A.area(lastx, mid);
    if (ar > target)
        hi = mid;
    else
        lo = mid;
    return lo;
}

public static long[] func_f8fd18658b6f48bb9c3ff06f4a807216(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
```

```
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    return counts;
}

public static int func_bd2761a1b3764cf8a4196f0aa3df0dc0(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    return x0;
}

public static int func_bc0e822187974ab5ac80929bb19835f7(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
```

```
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalf;
}

public static int func_1f67e69ed10a4372b08bd111a1688b45(int r, int p,
int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return s;
}

public static int[] func_a391aee98f5c48598f9060341102281c(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    return qty;
}

public static long func_cb03decebc664f3e9749c56cbf0d8b34(int n, long
right, long left, long[] a){
    long mid = (left + right) / 2;
    int last = 0;
    boolean ok = false;
    long sum1 = 0;
    int first = 0;
    while (first != n && sum1 < mid) {
```

```
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    return sum1;
}

public static double func_a59430bc400f44c9b031c1f72971c9c2(double k,
double remain, double x, double y){
    double a = k;
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    return d;
}

public static int func_41e4505a1ba4471a9ce9ff044b73fc21(int n, int
min, int h, long best, long[] imos){
    int max = n - 1;
    for (int cur = 0; cur < 40; cur++) {
        int med1 = (min * 2 + max) / 3;
        int med2 = (min + max * 2) / 3;
        long v1 = ProblemA.take(imos, h, med1);
        long v2 = ProblemA.take(imos, h, med2);
        if (v1 < v2) {
            min = med1;
        } else {
            max = med2;
        }
    }
    for (int c = min - 2; c <= max + 2; c++) {
        if (h <= c && c < n) {
            best = Math.max(best, ProblemA.take(imos, h, c));
        }
    }
    return max;
}

public static int func_881e1d416dca4746b973ddef7ca5ca3e(int n, int
first, long sum1, long mid, long[] a){
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    return second;
}
```

```java
public static int func_0bc7d6d744ab4f8ba4b088007f158faf(int n, int p,
Scanner sc){
    int q = sc.nextInt();
    int r = sc.nextInt();
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    return r;
}

public static double func_33359373698b4dfaafd741644d3816f8(double
target, double hi, double mid, double lastx, double lo){
    double ar = A.area(lastx, mid);
    if (ar > target)
        hi = mid;
    else
        lo = mid;
    return hi;
}

public static long func_fc5819a13611490bac0ad83cdd783f29(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return sum;
}
```

```
public static long func_029549866ab2463cbb935b4fb23cd645(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    return ss;
}

public static int[] func_dea9bd2a2d3444ba9340e4b3c508711f(Scanner
scanner){
    int n = scanner.nextInt();
    int p = scanner.nextInt();
    int q = scanner.nextInt();
    int r = scanner.nextInt();
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    return array;
}

public static double func_7d8fd422cbde4f998d725127dfea3758(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return dy1;
}

public static long func_cff42e2ab1264d90bd012bb543a5b99c(int n, int p,
int q, int r, Scanner sc){
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
```

```
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return tmp;
}

public static long func_5f0e6b9063dc4650b2385b3147b92496(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    return hi;
}

public static long func_3094f9e2847f4244a0d1c3b06a28b5c9(int lo, int
i, int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
```

```
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    return diff2;
}

public static long func_26ed6011c392476cb1652aff0aed51db(int n, long
left, long max, long center, long[] S){
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return right;
}

public static int func_4cc62db1069443e889a54cfcbcde46aa(int i, int j,
int[] p){
    int x = p[i];
    p[i] = p[j];
    p[j] = x;
    return x;
}

public static long func_4a0dfdff47de48d4af5c72014cfd8607(int joined,
long budget, long lower, long upper, long[] amount){
    long middle = (lower + upper) / 2;
    if (Cheaters.possible(amount, joined, middle, budget))
        lower = middle;
    else
        upper = middle;
    return upper;
}

public static double func_f2447edfcea84394a4fd867b46bd05f4(int l,
double bx, double ans, double ax, double ay){
    double by = A.highy[l + 1];
    if (ax > A.highx[l]) {
        ay = A.highy[l] + (ax - A.highx[l]) / (A.highx[l + 1] -
A.highx[l]) * (A.highy[l + 1] - A.highy[l]);
    }
    if (bx < A.highx[l + 1]) {
        by = A.highy[l + 1] + (bx - A.highx[l + 1]) / (A.highx[l] -
A.highx[l + 1]) * (A.highy[l] - A.highy[l + 1]);
    }
    ans -= (bx - ax) * (1000 - (ay + by) / 2);
    return by;
```

```
}

public static long func_45605d125c2f4e4a8d5fcdac9f71042b(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return sum;
}

public static long func_1bc914f97da343ec8cd3de56b528cdf9(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return ss;
}

public static long func_6af0c358df974cc2a8af5feb870d9954(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
```

```
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return sum;
}

public static long func_97555eabf37b4128a5dfdba805fb52dc(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
```

```
    return temp;
}

public static long[] func_2bca5f90e1f14b738cd702ad7f53408d(int cn, int
N, long p, PrintWriter out, Scanner in){
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] d = new long[N];
    for (int i = 0; i < N; i++) {
        d[i] = (p * i + q) % r + s;
    }
    out.println(String.format("Case #%d: %.11f", cn,
ProblemA.solve(d)));
    return d;
}

public static long[] func_47bde581f27947a1996c854267eea4d6(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    return partial;
}

public static long func_d7417206533742fa818bd009678c0389(int n, int
beforeHalf, long sum, long beforeHalfSum, long[] A){
    sum += A[n];
    while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum / 2)
{
        beforeHalfSum += A[beforeHalf];
        beforeHalf++;
    }
    long max = sum;
    if (beforeHalf > 0) {
        max = Math.max(beforeHalfSum, sum - beforeHalfSum);
    }
    return beforeHalfSum;
}

public static long func_7c071abc39d344a989c4a9a24cf59f51(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
```

```
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return sum;
}

public static long[] func_10fff2900c1b49a6a9b915ba447274b8(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    return A;
}

public static int func_d4ad9103fdd44151b94fdc89c9f37b46(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return index;
```

```
}

public static double func_996225415b83416cacf150d8136c8c9f(double
right, double dy1, double left, double dy2, double dx){
    double mid = (left + right) / 2.0;
    double dy2_ = (dy2 - dy1) * (mid / dx) + dy1;
    double S2 = (dy1 + dy2_) * mid / 2.0;
    return mid;
}

public static long[] func_3c35edc5ea544d1ca02dbd41333c3931(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return a;
}

public static double func_37bbe103a4884a839124f814bbfea4b8(int W){
    double max = W + 0.00000001337;
    double mid = 0;
    return mid;
}

public static double func_d7c993aa8dcd4e5e87fb133c34488623(int W, int
U, int G, int L, Scanner input){
    CodeJam_Round3_A.lower = new double[L][2];
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
```

```
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    return g;
}

public static long func_ab5bb5a4efa84f71ab9552f116ec1e2d(int i, int
right, int left, int max, int min){
    long leftCount = A.count(i, left);
    long rightCount = A.count(i, right);
    if (leftCount < rightCount) {
        max = right;
    } else {
        min = left;
    }
    return rightCount;
}

public static long func_431a67119e74479ca408c8de59f75ef8(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    return t1;
}

public static double func_659485958afb416ab3278c3d0e7e53f8(int W, int
G, int t){
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
```

```
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    System.out.println("Case #" + t + ":");
    return firstArea;
}

public static long[] func_c59d52d19de34b62932945a129d517af(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return best;
}

public static long[] func_d98221c18acc487ba5079c11fe6b0659(int N, long
p, long r, long s, long q){
```

```
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return arr;
}

public static int func_61b267cbeba8449c8440263d23b6c35c(int lo, int i,
int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    return ans;
}

public static long func_c45d66d1017045e5b649b9929a722223(int
winningThings, long payMoney, long low, long budget, long[] x){
    for (int i = 0; i < winningThings; i++) {
        payMoney += low - x[i];
    }
    if (payMoney > budget) {
```

```
        return -1;
    }
    --low;
    long high = budget + 1;
    while (high - low > 1) {
        long mid = low + high >> 1;
        if (A.canDo(budget, x, winningThings, mid)) {
            low = mid;
        } else {
            high = mid;
        }
    }
    if (low < x[winningThings - 1]) {
        return -1;
    } else {
        return low;
    }
}

public static double func_467685e8c50d4d49a6faa07ff073a860(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return crt;
}

public static double func_51a7ca64332242e7b538ed7a0d68e8a8(int
winningThings, long budget, long lowestBet, long payMoney, long[] x){
    if (payMoney > budget) {
        return -1;
    }
    long ourWin = 0;
    for (int i = 0; i < winningThings; i++) {
        ourWin += lowestBet - x[i];
    }
    return (double) ourWin / (double) winningThings * 36.0 - payMoney;
}

public static long[] func_46cf7357a78f4312abf6415c89363c93(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
```

```
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return cum;
}

public static double func_618e42cb1a5a4038a51902442d5a14d4(int l,
double ans, double ax, double right){
    double bx = Math.min(A.lowx[l + 1], right);
    double ay = A.lowy[l];
    double by = A.lowy[l + 1];
    if (ax > A.lowx[l]) {
        ay = A.lowy[l] + (ax - A.lowx[l]) / (A.lowx[l + 1] -
A.lowx[l]) * (A.lowy[l + 1] - A.lowy[l]);
    }
    if (bx < A.lowx[l + 1]) {
        by = A.lowy[l + 1] + (bx - A.lowx[l + 1]) / (A.lowx[l] -
A.lowx[l + 1]) * (A.lowy[l] - A.lowy[l + 1]);
    }
    ans -= (bx - ax) * ((ay + by) / 2 + 1000);
    return bx;
}

public static double func_4bb239c64c404b9683239be38ca1c989(double dy1,
double dy2, double dx, double S0){
    double left = 0.0, right = dx;
    for (int k = 0; k < 60; k++) {
        double mid = (left + right) / 2.0;
        double dy2_ = (dy2 - dy1) * (mid / dx) + dy1;
        double S2 = (dy1 + dy2_) * mid / 2.0;
        if (S2 < S0)
            left = mid;
        else
            right = mid;
    }
    return right;
}
```

```
public static long func_d6d7dca8badd4f898280c11a0874ecc6(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    for (int i = 0, j = n - 1; i <= j; ) {
        if (si + a[i] < sj + a[j]) {
            si += a[i++];
        } else {
            sj += a[j--];
        }
        min = Math.min(min, Tour.max(sum - si - sj, si, sj));
    }
    return si;
}

public static double func_f2a3ce33aced4be4885a56adf86e018c(double
target, double lastx, double lo){
    double hi = A.W;
    double mid = 0;
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    return hi;
}

public static long func_d584f709e95b40a5b53f4a4812965fd6(int
beforeHalf, long max, long sum, long beforeHalfSum, long[] A){
    long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
    max = Math.min(max, value);
    return max;
}

public static int func_74423c99f9254c6c995638e3e5aa04ef(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
```

```
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return first;
}

public static int[] func_f825809d0aeb4ccd8d528d0725dc7286(Scanner s){
    int N = s.nextInt();
    int[] len = new int[N];
    int[] prob = new int[N];
    for (int i = 0; i < N; i++) {
        len[i] = s.nextInt();
    }
    return prob;
}

public static long[] func_2cff9b46bdd746af86a4cce56fc77c03(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return dp;
}

public static long func_cb45d35c59dd4853a0825c3bdbba5581(int n, int b,
int a, int i, long[] sum){
    long s1 = sum[b - i + 1] - sum[a];
```

```
    long s2 = sum[a];
    long s3 = sum[n] - sum[b - i + 1];
    long smax = Math.max(s1, Math.max(s2, s3));
    return s1;
}

public static long func_ee89497f17824997b104b95d051da6e2(int numtie,
int i, long bet, long waste, long dif){
    waste += numtie * dif;
    bet += i * dif;
    return waste;
}

public static long[] func_d5e3df8c612947bb8f1d62d41514a3b8(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    return rsum;
}

public static long[] func_3adca19ca1b541b7af7efc7b99543d7a(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    long total = partials[n];
    return partials;
}
```

```java
public static long func_e6076e6a017644c5a4ba57ced47b85cd(int numtie,
int i, long bet, long waste, long dif){
    waste += numtie * dif;
    bet += i * dif;
    return bet;
}

public static int func_7d0ceffdeb12440e8c905f22f24ecb99(int lo, int i,
int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    return lo;
}

public static int func_9a13c362ce3b47a7aae9ac31091941b4(int n, int[]
ind, int[] l, int[] d){
    for (int i = 0; i < n; ++i) {
        for (int j = i + 1; j < n; ++j) {
            if ((l[ind[i]] * d[ind[i]] < l[ind[j]] * d[ind[j]]) ||
(l[ind[i]] * d[ind[i]] == l[ind[j]] * d[ind[j]] && (d[ind[i]] == 0 &&
ind[i] > ind[j] || (d[ind[i]] != 0 && (l[ind[i]] > l[ind[j]] ||
(l[ind[i]] == l[ind[j]] && ind[i] > ind[j])))))) {
                int t = ind[i];
                ind[i] = ind[j];
                ind[j] = t;
            }
        }
    }
    final int result = 0;
    return result;
}

public static long func_66954f2c1a0f44d69aaefd56d70cf933(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
```

```
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    return hi;
}

public static long[] func_0e683e86a8624a80976c588532034f7c(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return A;
}

public static double func_82f32aafb6f1427cba5cc378c52cc7d8(double t,
double k, double res, double d1, double dx){
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return res;
}

public static long[] func_fd1f9179ee3640e4bb35563686aaed63(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    return partial;
}

public static double func_43d2d640f5f24eb59018a511f2910791(double
target, double lastx, double lo){
    double hi = A.W;
    double mid = 0;
```

```
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    return lo;
}

public static long[] func_0cdb8bdd5053440086054d64ee474bea(int n, long
p, long q, Scanner br){
    long r = br.nextInt();
    long s = br.nextInt();
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return nums;
}

public static double func_794e156f74034fc2acd1fc0b3ea03d15(double t,
double k, double d1, double want, double dx){
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
    }
    return temp;
}

public static int func_3fc13be69a6f4fd589447f8860bc013e(A2.Pair p2,
A2.Pair p1){
    int x0 = p1.a;
```

```
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return x1;
}

public static long func_b43b74fdcea74502ac04578d7507b23b(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return sj;
}

public static long[] func_de82a3d16e5c4973bb266f7427991078(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return A;
}
```

```
public static long func_7a900b9d36c34b2cbe029c9f3ca313fa(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return beforeHalfSum;
}

public static double func_3c6fc5e536b1401badbbf0994f39cab7(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
```

```
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    double res = (A[N] - ans) * 1.0 / A[N];
    return res;
}

public static long[] func_4d39d1f6bdb14588a54925cf29e75f74(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    return a;
}

public static long[] func_7690315601244bbbaf9cd64f731e4d96(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
```

```
    }
    return best;
}

public static long func_dc8cbbea92ab420d8ff8242817d27127(int n, int
first, long sum1, long mid, long[] a){
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
        second--;
    }
    return sum1;
}

public static double func_4e7c8fd5c9d84d0ead90fbf07d9b47ef(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return dy0;
}

public static long[] func_5ccd2c0502a34a0591b2889a6bbf8bbc(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return dp;
}

public static long[] func_428f6ea1b66f4e99b93e8165de89d956(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    return nums;
```

```
}

public static long func_e980e51ec2b043b88bd6c1956ee5e613(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return ans;
}

public static long func_628e25afe7e340e8b27a780589753112(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
```

```
    return lo;
}

public static long func_75ff805ae5324deda229924741baf5a0(int r, int s,
int q, int i, long curr){
    curr += q;
    curr %= r;
    curr += s;
    A.num[i] = A.num[i - 1] + curr;
    return curr;
}

public static int[] func_bb4b4c25d9f24704b6834b9641ca15b7(Scanner sc){
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    int[] xl = new int[l];
    int[] yl = new int[l];
    int[] xu = new int[u];
    int[] yu = new int[u];
    for (int i = 0; i < l; ++i) {
        xl[i] = sc.nextInt();
        yl[i] = sc.nextInt();
    }
    for (int i = 0; i < u; ++i) {
        xu[i] = sc.nextInt();
        yu[i] = sc.nextInt();
    }
    int[] xs = new int[l + u];
    return xs;
}

public static double func_d7f7fc9cf5a94a4ba30b63f731067257(double
target, double hi, double mid, double lastx, double lo){
    mid = (hi + lo) / 2;
    double ar = A.area(lastx, mid);
    if (ar > target)
        hi = mid;
    else
        lo = mid;
    return hi;
}

public static int[] func_1902c90beef9484aa2a2925149ce207e(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
```

```
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    return arr;
}

public static long func_a48fb06c141545318e7ea1255af83b8c(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    return sum;
}

public static int func_8487c698fa194d23ac8f3e5da460a30a(int y0, int
x0, int x, double[] low, Scanner in){
    int y = in.nextInt();
    double k = 1.0 * (y - y0) / (x - x0);
    for (int j = x0 + 1; j <= x; j++) {
        low[j] = k * (j - x0) + y0;
    }
    x0 = x;
    return x0;
}

public static int func_354b06495d384e2398a579abc2324d92(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    int Tba = 100 * Lb + La * (100 - Fb);
    return Tba;
}

public static double func_7b40943037a74fcf9755699710fdf800(int i, long
bet, long waste){
    long v1 = 36 * bet;
    double val = v1 / (i + 0.0);
    val -= bet;
    val -= waste;
    System.out.println(val);
```

```
    return val;
}

public static long[] func_288182d07aa04450b7a2716775f31dd1(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    long best = Long.MAX_VALUE;
    return a;
}

public static long func_cb43e6c79d1842de8688ff268943f202(int n, int
first, long sum1, long mid, long[] a){
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
        second--;
    }
    return sum1;
}

public static long func_f656b9522b36401881a673ac6af1df38(int
winningThings, long payMoney, long low, long budget, long[] x){
    for (int i = 0; i < winningThings; i++) {
        payMoney += low - x[i];
    }
    if (payMoney > budget) {
        return -1;
    }
    --low;
    return payMoney;
}

public static long[] func_fb797bbe55a542b491e90c31f7430f2e(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
```

```
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    long max = 0;
    return amt;
}

public static int func_6f30ddbc83b748a5b8ec2da23e8219a7(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return first;
}

public static long func_38cd67eec7f54fe1a832da808f61e324(int lo, int
i, int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
```

```java
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    return left;
}

public static long func_25022bba79d449ab9b9b51d58618de62(int numtie,
int i, long waste, long initadd, long[] v){
    if (v[i] == v[i - 1]) {
        v[i]++;
        initadd++;
        waste += numtie;
    }
    long bet = i * (v[i] - 1);
    long min = 0;
    return min;
}

public static int[] func_1c3f0718fbe64e908216230567c6ba9d(int r, int
p, int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    return A;
}

public static long func_bbf43ea822e244a18a630561650c60d3(int mid, int
n, long prefixSum, long[] rsum){
    long sum = (rsum[mid] - prefixSum);
    long suffSum = rsum[n - 1] - sum - prefixSum;
    return sum;
}

public static double[] func_3b63487feb464d7cb90b457a617cc342(int w,
int l, Scanner in){
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
```

```
        x0 = x;
        y0 = y;
    }
    return len;
}

public static double func_c81b0ed1f44a49af8edb5a1aedb301c3(double x,
A.Point st, A.Point en){
    double coef = (x - st.x) / (en.x - st.x);
    return st.y + coef * (en.y - st.y);
}

public static long func_973b20e67cb146d386a59333e55a9953(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return min;
}

public static long func_a4d85eb8d1b34a12a43deeff5802040e(int n, long
P, long Q, Scanner in){
    long R = in.nextInt();
```

```
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return S;
}

public static long[] func_0a4e6d3ccccb4494938f9526aaedbf48(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return arr;
}

public static double func_28d01c13af704cfeb9ae6b83c2ca578e(double x,
Segment upperSegment){
    double left = upperSegment.a.x;
    double right = x;
    return left;
}

public static long func_27f5e23f569e47eea0c752aea599b7bf(long budget,
long payMoney){
    if (payMoney > budget) {
        return -1;
```

```
    }
    long ourWin = 0;
    return ourWin;
}

public static long func_03cd9e5069af4ab098ce1c1d8bd07084(int n, int[]
values, long sum){
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return middle;
}

public static long func_29eb8841bb6d433fa74c153784288d67(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
```

```
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return sum;
}

public static long[] func_627271ee627f4ab098f5363e5b2bc9ac(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    for (int a = 0, b = 0; a < n; a++) {
        while (sum[b + 1] - sum[a] < sum[n] - sum[b + 1] && b < n) {
            b++;
        }
        for (int i = 0; i < 2; i++) {
            if (b - i < a)
                break;
            long s1 = sum[b - i + 1] - sum[a];
            long s2 = sum[a];
            long s3 = sum[n] - sum[b - i + 1];
            long smax = Math.max(s1, Math.max(s2, s3));
            res = Math.max(res, 1 - (double) smax / sum[n]);
        }
    }
    return sum;
}

public static long func_6effab03d4264fd194ba989a0096578c(int n, long
right, long left, long[] a, long[] sum){
```

```
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    return right;
}

public static double func_9a023eb483da43dfb321ae51990c140e(int l, int
g, int u, int w, int[][] pos){
    double[] bound = new double[w + 1];
    int px = pos[0][0];
    int py = pos[0][1];
    for (int i = 1; i < l; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] -= (ny - py) / div * (j - px) + py;
        }
```

```
        px = nx;
        py = ny;
    }
    px = pos[l][0];
    py = pos[l][1];
    for (int i = l + 1; i < l + u; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == l + 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] += (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    double sum = 0.0;
    for (int i = 0; i < w; ++i) {
        sum += (bound[i] + bound[i + 1]) / 2.0;
    }
    double req = sum / g;
    double cur = 0.0;
    int p = 0;
    double used = 0.0;
    return sum;
}

public static long func_93b79919314e47129ddb49802a6c8182(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return s2;
}

public static long func_6008712b84a64dd4978a54d506fa1771(int n, int b,
int a, int i, long[] sum){
    long s2 = sum[a];
    long s3 = sum[n] - sum[b - i + 1];
    return s3;
}

public static long func_1094cdd871764dc1a07c84bf5e6f21ee(int numtie,
int i, long bet, long waste, long dif){
    waste += numtie * dif;
    bet += i * dif;
    return bet;
}
```

```java
public static long func_02558f02ef9b4af69fa7e24eede0844b(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return tot;
}

public static long func_e4f70c53934941db8f0d90459a197892(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    return sum;
}

public static long func_e04cde2f6ae047abb670fbb22dc21bc9(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    return sum;
}

public static long[] func_c4b945b563cc4eecab34a438ffdf8f41(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
```

```
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return arr;
}

public static long func_f2b2168d78a747b0aa05954502b03888(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return HI;
}

public static long[] func_97ae6caa358d48f591bc889eff617134(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
```

```
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    return arr;
}

public static long func_37bdbbe06cc5458bb43357b8e69870de(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return t2;
}

public static int[] func_f2b0ef457a414e85a3a3f54dc94670c2(int r, int
p, int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    for (int i = 0; i < n; i++) {
        int lo = 0;
        int hi = i;
        int ans = -1;
        long tot = S[i + 1] - S[0];
```

```
        while (lo <= hi) {
            int mid = (lo + hi) / 2;
            long center = S[i + 1] - S[mid];
            long left = tot - center;
            // System.out.println(mid + " " + left + " " + center);
            if (left > center) {
                hi = mid - 1;
            } else {
                ans = mid;
                lo = mid + 1;
            }
        }
        // System.out.println(i + " " + ans);
        long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
        long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
        if (diff2 < diff1)
            ans++;
        // System.out.println(i + " " + ans + " " + diff1 + " " +
        // diff2);
        long center = S[i + 1] - S[ans];
        long left = S[ans] - S[0];
        long right = S[n] - left - center;
        long take = S[n] - Math.max(center, Math.max(left, right));
        max = Math.max(max, take);
    }
    return A;
}

public static int func_8fcca433207d47c39cced68b9c86459a(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
```

```
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return beforeHalf;
}

public static long func_382a97879ec94092b7e0acb05ef86c88(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return s;
}

public static long[] func_6545830ab58143d49aa76d87f0b9b27a(int c, int
n, long p, long q, Scanner br){
    long r = br.nextInt();
    long s = br.nextInt();
```

```
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    for (int i = 0; i < n; i++) {
        int low = i;
        int high = n - 1;
        long prefixSum = (i == 0 ? 0 : rsum[i - 1]);
        while (high - low > 1) {
            int mid = (high + low) / 2;
            long sum = (rsum[mid] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            if (sum <= suffSum) {
                low = mid;
            } else {
                high = mid;
            }
        }
        for (int j = low; j <= high; j++) {
            long sum = (rsum[j] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            long max = Math.max(sum, suffSum);
            max = Math.max(max, prefixSum);
            best = Math.max(best, rsum[n - 1] - max);
        }
    }
    System.out.printf("Case #%d: %.10f\n", c, (best * 1.0) / rsum[n -
1]);
    return nums;
}

public static long func_c3c0be3a39fe4423b17e6f9f7f9aaf41(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
```

```
        tot += arr[i];
    }
    long L = 0;
    return tot;
}

public static double func_5e86c73653504fd6af1a72e6fd107b05(int n, long
t2, long sum, long t1, long[] a){
    long rest = sum;
    int j = 0;
    double ans = 0;
    for (int i = 0; i < n; i++) {
        while (j < n && t2 + a[j] <= rest - a[j]) {
            t2 += a[j];
            rest -= a[j];
            j++;
        }
        double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) /
sum;
        ans = Math.max(ans, pp);
        if (j < n) {
            pp = 1.0 * (sum - Math.max(t1, Math.max(t2 + a[j], rest -
a[j]))) / sum;
            ans = Math.max(ans, pp);
        }
        t1 += a[i];
        t2 -= a[i];
        if (j == i) {
            rest -= a[j];
            j++;
        }
    }
    return ans;
}

public static double[] func_d425c63a19564692b8e88fa628a93fb7(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
```

```
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    return low;
}

public static long[] func_333b50491c7642e5a0771c402aa45b90(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
```

```
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return A;
}

public static long func_749dc745a33e4764863390701a86817b(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return middle;
}

public static long func_9fcfa598d4944f449170c5f4ee8c3b94(int N, long
q, long p, long r, long s){
```

```
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return beforeHalfSum;
}

public static double func_6991281044d64983b588f4f8faa18d04(int r, int
s, int n, int p, int q){
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
```

```
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    return ret;
}

public static int func_c285ed1432f34e0198710d5191cb7217(int r, int p,
int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    return s;
}

public static long func_71eb0a8f7824431793f2fec3aa6e43ca(int n, int p,
int q, int r, Scanner sc){
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    return sum;
}

public static long func_44bdf5e181eb4d4296e61bdf34f5e45b(int s, int N,
int q, int p, int r){
    long[] n = new long[N];
    for (int i = 0; i < N; i++) {
        n[i] = ((long) i * p + q) % r + s;
        if (i > 0) {
            n[i] += n[i - 1];
        }
    }
    long ans = Long.MAX_VALUE / 4;
    return ans;
}

public static double func_417350743033433e83a5fa2736c5622b(double min,
```

```
double max){
    double mid = (min + max) / 2;
    double[] poly = new double[8];
    poly[0] = mid;
    poly[1] = 1001;
    return mid;
}

public static long func_be2a963cefe1436e81dbaff547ebc2c8(int n, int s,
int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    return left;
}

public static int[] func_65003d72612a4591a835873521c91b99(int n, int
r, int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
```

```
        middle -= values[a];
    }
    return values;
}

public static int[] func_467bcb76078a4c7f9ef524c2b3bd1c53(int l, int
u, int[] lx, int[] ly, Scanner input){
    for (int i = 0; i < l; i++) {
        lx[i] = input.nextInt();
        ly[i] = input.nextInt();
    }
    int[] ux = new int[u];
    int[] uy = new int[u];
    return ux;
}

public static long[] func_f8f4b59f7e574e9a8cddb16e0446d2a0(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    return a;
}

public static long[] func_9bcbfff55dc54f5bb667f105e4dea502(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    return pref;
}

public static int[] func_9d56acafc85f46b4bbb6fde7abccc6eb(int r, int
p, int q, int n, Scanner in){
    int s = in.nextInt();
```

```
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    for (int i = 0; i < n; i++) {
        int lo = 0;
        int hi = i;
        int ans = -1;
        long tot = S[i + 1] - S[0];
        while (lo <= hi) {
            int mid = (lo + hi) / 2;
            long center = S[i + 1] - S[mid];
            long left = tot - center;
            // System.out.println(mid + " " + left + " " + center);
            if (left > center) {
                hi = mid - 1;
            } else {
                ans = mid;
                lo = mid + 1;
            }
        }
        // System.out.println(i + " " + ans);
        long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
        long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
        if (diff2 < diff1)
            ans++;
        // System.out.println(i + " " + ans + " " + diff1 + " " +
        // diff2);
        long center = S[i + 1] - S[ans];
        long left = S[ans] - S[0];
        long right = S[n] - left - center;
        long take = S[n] - Math.max(center, Math.max(left, right));
        max = Math.max(max, take);
    }
    return A;
}

public static long func_e38b16b3a77d453797d9c8383b670a5d(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
```

```
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return avg;
}

public static long[] func_ef34c580dd754c1780435cd9d2c99409(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return a;
}

public static long func_92a8c20958464e67a1314fd649f40e32(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return si;
}

public static long func_254a95334a0743e2902d4ade7e058e90(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    return l;
}

public static long[] func_8ca183b983484660bcfb02572171e6d1(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
```

```
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    return partials;
}

public static int[] func_80ae9cf7bdc7483dbf193c7d7ca09a90(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    return arr;
}

public static long[] func_532961ace5324bc99d179ba4a84b79eb(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    long[] pref = new long[n];
    for (int i = 0; i < n; i++) {
        pref[i] = ar[i];
        if (i > 0) {
            pref[i] += pref[i - 1];
        }
    }
    long max = Long.MAX_VALUE;
    return ar;
}

public static long func_8279f911ee3d48f3a21162f2c7a6b613(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
```

```
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return sum;
}

public static long func_b6e32c9df2e54fb79e16b6e782b26bfc(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    while (pow2 * 2 <= N) {
        pow2 *= 2;
    }
    return LO;
}

public static long func_8fa4e28d2aae437e902f8b9a9c30ecdb(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
```

```
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    return temp;
}

public static double func_4b5210384a414d20a9c5a58fede6c0c4(int i,
double[] y, double[] x, double[] angleUp, double[] up){
    double Y = (y[i] - y[i - 1]) / (x[i] - x[i - 1]);
    double z = y[i - 1] + Y;
    for (int j = (int) x[i - 1] + 1; j <= x[i]; j++) {
        up[j] = z;
        z += Y;
        angleUp[j] = Y;
    }
    return Y;
}

public static long func_c7ec811203d2485cb632b4ffde236a43(int numtie,
int i, long bet, long waste, long dif){
    waste += numtie * dif;
    bet += i * dif;
    numtie++;
    return waste;
}

public static double func_047bfbc25a8046a1b6c05dd6efe6f566(double ku,
double st, double kl, double scur){
    double sss = st - scur;
    double a = ku - kl;
    return sss;
}

public static long func_814943ca852d4f1d956776714bca288b(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
```

```
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return sum;
}

public static long func_732a8fe00d434302982c274fc4344ae0(long x, long
best, long sum, Long down){
    long now = Math.max(down, Math.max(x - down, sum - x));
    best = Math.min(best, now);
    return best;
}

public static long func_6592865370ba4609ae30e4e040f7a8b7(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return rest;
}

public static double func_86793bcfa97e47db9e52f26eca47d087(int n, int
x, int y, long sum, long[] a){
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
```

```
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return ans;
}

public static int[] func_7d53d43fc13e48359b1896c1b2d874ab(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    return a;
}

public static double func_909a6e9865d94df1bc8041018b63d3f1(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    double ans = 0;
    for (int i = 0; i < n; i++) {
```

```
        for (int j = i + 1; j <= n; j++) {
            int s1 = ps[i];
            int s2 = ps[j] - ps[i];
            int s3 = ps[n] - ps[j];
            int max = Math.max(s1, Math.max(s2, s3));
            ans = Math.max(ans, 1.0 * (ps[n] - max) / ps[n]);
        }
    }
    return ans;
}

public static long[] func_2e960bc6a3c443198d27a6f078ac17e0(int r, int
p, int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return S;
}

public static long[] func_ab2131ae8a7543f685e80b8ac97930ed(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    return A;
}

public static int[] func_ef9e0d484bef4a6e89e135dc39b3d4b4(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
```

```
    // sout(i + " " + index);
    }
    return a;
}

public static long func_1caa105e98cc433ba387dc9f07dfcaed(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return rest;
}

public static long[] func_73761caef77349a29c5b240deb5bd82b(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    return sum;
}

public static double func_7d601b601873468ca48920d14ce0af15(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
```

```
    }
    double ans = 0;
    return ans;
}

public static long func_f1f78d368307436cb16900592ebd074a(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return t2;
}

public static int func_d3c46994d9db4f14a5f6772f1c0497a4(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return second;
}

public static int func_fd864ea73be3411ab9af74427c88cdf3(int u, int l,
int tt, int g, Scanner in){
    int[] x1 = new int[l], y1 = new int[l], x2 = new int[u], y2 = new
int[u];
    for (int i = 0; i < l; i++) {
```

```
            x1[i] = in.nextInt();
            y1[i] = in.nextInt();
        }
        for (int i = 0; i < u; i++) {
            x2[i] = in.nextInt();
            y2[i] = in.nextInt();
        }
        double area = 0;
        for (int i = 0; i < l - 1; i++) {
            area += x1[i] * y1[i + 1] - x1[i + 1] * y1[i];
        }
        area += x1[l - 1] * y2[u - 1] - x2[u - 1] * y1[l - 1];
        for (int i = u - 1; i >= 1; i--) {
            area += x2[i] * y2[i - 1] - x2[i - 1] * y2[i];
        }
        area += x2[0] * y1[0] - x1[0] * y2[0];
        area = Math.abs(area) / g;
        int p1 = 0, p2 = 0;
        double cx = 0;
        System.out.println("Case #" + tt + ":");
        for (int i = 0; i < g - 1; i++) {
            double ca = area;
            while (ca > 1e-9) {
                // System.out.println (p1 + " " + p2 + " " + ca);
                double nx = Math.min(x1[p1 + 1], x2[p2 + 1]);
                double a1 = y1[p1 + 1] - y1[p1];
                double b1 = x1[p1] - x1[p1 + 1];
                double c1 = a1 * x1[p1] + b1 * y1[p1];
                double a2 = y2[p2 + 1] - y2[p2];
                double b2 = x2[p2] - x2[p2 + 1];
                double c2 = a2 * x2[p2] + b2 * y2[p2];
                for (int j = 0; j < 100; j++) {
                    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
                    cy1 = (c1 - a1 * cx) / b1;
                    my1 = (c1 - a1 * mx) / b1;
                    cy2 = (c2 - a2 * cx) / b2;
                    my2 = (c2 - a2 * mx) / b2;
                    double la = (cy2 - cy1 + my2 - my1) * (mx - cx);
                    // System.out.println (" --> " + la + " " + cx + " " +
mx  + " " + cy1 + " " + cy2 + " " + my1 + " " + my2);
                    if (la < ca) {
                        cx = mx;
                        ca -= la;
                    } else {
                        nx = mx;
                    }
                }
                if (ca > 1e-9) {
                    if (x1[p1 + 1] < x2[p2 + 1])
                        ++p1;
```

```
                else if (x1[p1 + 1] > x2[p2 + 1])
                    ++p2;
                else {
                    ++p1;
                    ++p2;
                }
            }
        }
        System.out.println(cx);
    }
    return p2;
}

public static long func_018ac2b0f0714ba981f4a2e32f332f98(int r, int p,
int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    return best;
}

public static int func_6572d800222e4f50bee9a675e693499b(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return j;
}

public static long func_3938c9c3e76742df8f5f664098d73c98(int r, int s,
```

```
int q, int i, long curr){
    curr += q;
    curr %= r;
    curr += s;
    A.num[i] = A.num[i - 1] + curr;
    return curr;
}

public static int[] func_0a8966d8b5644d9b843f05c7e4aa57eb(int r, int
p, int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return A;
}

public static double func_273caaf13bc44ad98097b72da7655235(double y0,
double total, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    for (int i = 0; i < lx.size(); ++i) {
        double xx = lx.get(i);
        double yy = ly.get(i);
        total += (x0 + xx) * (y0 - yy);
        x0 = xx;
        y0 = yy;
    }
    return x0;
}

public static long func_b29a158447b14a87b3962e782ef2680b(int to, long
value, long res, long[] a){
    double ss = res * 1.0 / to * 36;
    for (int i = to; i < 37; i++) if (a[i] <= value) {
        res += value + 1 - a[i];
    }
    ss -= res;
    return res;
}

public static long[] func_f3681e2706c4496dadde975c2fb48c70(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
```

```
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    return pref;
}

public static long[] func_a09966b4fb124510a9dc951d94b21400(int r, int
p, int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    long total = partials[n];
    return partials;
}

public static long func_d1f29565573f4819930714df2cd8643e(int p, int q,
int n, int r, int s){
    TaskA.a = new long[n];
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    for (int i = 2, j = 1; i < n; i++) {
        while (j + 1 < i && TaskA.sum(0, j + 1) <= TaskA.sum(j + 1,
i)) {
            ++j;
        }
        minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i, n),
TaskA.sum(j, i)), TaskA.sum(0, j)));
        if (j + 1 < i) {
```

```
            minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i,
n), TaskA.sum(j + 1, i)), TaskA.sum(0, j + 1)));
        }
    }
    return minimal;
}

public static long func_955ebb5ff9454576ad041feb8c21af87(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    return tmp;
}

public static long func_c4091c8c2553413db37db3e720d1701e(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    long max = 0;
    for (int i = 0; i < N; i++) {
        int lo = i + 1, hi = N;
        long left = psum[N] - psum[i];
        while (lo < hi) {
            int mid = (lo + hi) >> 1;
            long s1 = psum[mid] - psum[i];
            long s2 = left - s1;
            if (s1 >= s2) {
                hi = mid;
            } else {
                lo = mid + 1;
            }
        }
        long t = psum[lo] - psum[i];
        long a = Math.max(t, left - t);
        if (lo > i + 1) {
            t = psum[lo - 1] - psum[i];
            a = Math.min(a, Math.max(t, left - t));
        }
```

```
        long get = psum[N] - Math.max(psum[i], a);
        if (get > max)
            max = get;
    }
    return max;
}

public static long func_1e1c2dd2acc3463b9c9dd58756bb7a86(long sr, long
sm, long sl, long[] a){
    long ans = Math.max(sm, sr);
    int l = 0;
    for (int r = 1; r < a.length; ++r) {
        sm += a[r];
        sr -= a[r];
        while (l < r && Math.max(sl + a[l], sm - a[l]) < Math.max(sl,
sm)) {
            sl += a[l];
            sm -= a[l];
            ++l;
        }
        ans = Math.min(ans, Math.max(sr, Math.max(sm, sl)));
    // System.out.println(" " + l + " " + r + " " + sl + " " + sm + "
" + sr + " " + ans);
    }
    return sl;
}

public static int func_ae24bbbcfe7a4565a3425b0418eaa467(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return hi;
}

public static long func_14a033a2d8894864b00fbc84d1539e92(int p2, int
p3, int l2, int l3, long l1){
    long eCur = (l1 + l2) * p2 + (l1 + l2 + l3) * (100 - p2) * p3;
    long eSwap = (l1 + l3) * p3 + (l1 + l2 + l3) * (100 - p3) * p2;
    return eCur;
}

public static long[] func_7a447d0c87114d5fa1853f7ab4b4a15c(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
```

```
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return arr;
}

public static int[] func_bf230de7ec794ac6b4bd61eec844ac95(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    return array;
}

public static long func_6498148f9c4d4e48b3ccaa45a54fb5f0(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
```

```
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return temp;
}

public static long[] func_4a9e7c4e5e124332bc45cd03406636bb(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
```

```
                l = m;
            }
        }
    }
    return a;
}

public static long[] func_ed1d07ac1a9f43369055854e627d6a34(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
```

```
        }
    }
    Locale.setDefault(Locale.US);
    return a;
}

public static double func_83fbc428126b41678efe81079fa214ba(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return my1;
}

public static int func_97d10e33237b4a1a9825b0fb44b044a0(int n, Scanner
scanner){
    int p = scanner.nextInt();
    int q = scanner.nextInt();
    int r = scanner.nextInt();
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    return q;
}

public static long[] func_e1b8b329e38949a0905032e45bc93b80(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    return sum;
}

public static long func_73506b2f09ed4e719b9d9ac2619a7f65(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
```

```
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return LO;
}

public static long func_2015152a295547d4a3785ec5733ccb96(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    return ans;
}

public static int func_ec02001dc75a41d3b1ae8f0a5bff7db7(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return second;
}

public static long func_07750557025947d8a1951e3e814a952a(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
```

```
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    return tot;
}

public static long[] func_9dd4d60660ba46feb6f41083fd6501a1(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
```

```
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    return a;
}

public static int func_3a3ab63d615d47eba37ec43716a165cc(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    int Tba = 100 * Lb + La * (100 - Fb);
    return Fb;
}

public static int func_274ad6cf891546d8b3c63287ad330a1a(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return first;
}

public static double func_28423f0502824549b777d6457f79d7c0(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return crt;
}

public static int func_2f00ecc6c2d84dffa18db9978c61a379(int n, Scanner
in){
    long[] a = new long[n];
    int nn = in.nextInt();
```

```
    for (int i = 0; i < nn; i++) {
        a[i] = in.nextLong();
    }
    Arrays.sort(a);
    return nn;
}

public static long func_cb860067bba644ef9c898be2e000f196(int r, int s,
int p, int q, int i){
    long curr = i - 1;
    curr *= p;
    curr += q;
    curr %= r;
    curr += s;
    return curr;
}

public static long func_cf43453569d04cafb4e662bc6d1fc393(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return sum;
}

public static long func_b39e83d651614eb1a6bfef6150024322(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
```

```
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return beforeHalfSum;
}

public static double func_a216d0aee714471bb8b94fea3be40b6d(double X,
double[] angleDown, double[] angleUp, double[] down, double[] up){
    double upY = (X - (int) X) * angleUp[(int) X + 1] + up[(int) X];
    double downY = (X - (int) X) * angleDown[(int) X + 1] + down[(int)
X];
    double St = (upY - downY + up[(int) X + 1] - down[(int) X + 1]) *
((int) X + 1 - X) / 2;
    return downY;
}

public static long func_ce163dade2f74db297bd4c6ddfd9fb99(int i, int
ans, int n, long max, long[] S){
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return take;
}

public static int func_f78a79d421c44069a42d4b4c51b98083(int l, int g,
int u, int w, int[][] pos){
    double[] bound = new double[w + 1];
    int px = pos[0][0];
    int py = pos[0][1];
```

```
    for (int i = 1; i < l; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] -= (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    px = pos[l][0];
    py = pos[l][1];
    for (int i = l + 1; i < l + u; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == l + 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] += (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    double sum = 0.0;
    for (int i = 0; i < w; ++i) {
        sum += (bound[i] + bound[i + 1]) / 2.0;
    }
    double req = sum / g;
    double cur = 0.0;
    return px;
}

public static long[] func_550b0ecd21fd4026a0dd857c55a05bcc(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    return partials;
}

public static long func_b0577e91802746599c4c601c95014e08(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
```

```
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    return middle;
}

public static double func_4fa2f098040244a8a5a5ca26a2986295(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return k;
}

public static long func_dcfe89918aaa4913ae6c004994ff6e2a(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return sum;
}

public static double[] func_e83db9bd090f487e946389760942d1e8(int t,
int w, Scanner in){
    int l = in.nextInt();
    int u = in.nextInt();
```

```
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    return len;
}

public static int func_70efa4fc645b4ff8a1cc2962bddfc7ee(int n){
    int p = MagicalTour.input.nextInt(), q =
MagicalTour.input.nextInt(), r = MagicalTour.input.nextInt(), s =
MagicalTour.input.nextInt();
    int[] values = new int[n];
    return s;
}

public static long func_c06738243d4f42338cbb71d8b5882be4(int N, long
min, long[] best, long[] A){
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return min;
}

public static double func_639c6557d85e484f83d2577e3539090f(int W, int
U, int L, int t, Scanner input){
    int G = input.nextInt();
    CodeJam_Round3_A.lower = new double[L][2];
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
```

```
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    System.out.println("Case #" + t + ":");
    return g;
}

public static long[] func_9bd8ecc007f14b45bca64b2af299e154(int N){
    long[] arr = new long[N];
    long[] cum = new long[N];
    return cum;
}

public static long func_288e5a8fd4214129b66d040bbff57324(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    return hi;
}

public static double func_374dfe9470224af58a4d7c88207dcdd9(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
```

```
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    lastx = mid;
    return lastx;
}

public static long[] func_2cb0ec3729224e58ac29a92f02db3bce(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    return pref;
}

public static int func_98a48f1d63a5479495eb129f29550f30(int r, int p,
int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    return s;
}

public static double func_93a1199736df4d659344de02924e5abf(double a,
```

```
double c, double b){
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    double x3 = -1.0 * (d + b) / (2 * a);
    return x2;
}

public static double func_1fb8675077e14e149985c89cd7d290ca(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return cy1;
}

public static long func_dbba45c511f741f19e6f08fe5bacdf36(int r, int s,
int i, long curr){
    curr %= r;
    curr += s;
    A.num[i] = A.num[i - 1] + curr;
    return curr;
}

public static long func_08446310b44f48b4a2554251a750c512(int N, int t,
long p, PrintWriter out, Scanner in){
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
```

```
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    out.println("Case #" + (t + 1) + ": " + answer);
    return sum;
}

public static int func_f14daa2579dd42c7a28641927daa5455(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    return Tab;
}

public static long func_ec62618649a241a98ea8d921b5751070(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return rest;
}

public static double func_e903438b01c1444bbb5deef40b5d7e07(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
```

```
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
    }
    return k;
}

public static long[] func_8ace51993a604df28877376214bb7584(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return A;
}

public static double func_aa0b8988dafc4ad4a2349534edea3d3e(int W, int
U, int G, int L, Scanner input){
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
```

```
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    return g;
}

public static int func_bb23ddbfac654fc18e254ef6f37acd85(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return first;
}

public static long func_53a3f1f5cbc3487ca08fc19b3c967fc7(int r, int p,
int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return max;
}

public static int func_4413827c0d6e4a1eae3383f911c2e39d(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return hi;
}

public static long[] func_9439a95adbff45789f6623701441ab5a(int r, int
```

```
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    return a;
}

public static int func_8436d1e67a9b4b1aa1127145c004c41c(int rght, long
rs, long sum, long max, long[] pref){
    int left = -1, right = rght;
    while (left < right - 1) {
        int mid = (left + right) >> 1;
        if (pref[mid] * 2 >= sum) {
            right = mid;
        } else {
            left = mid;
        }
    }
    for (int t = right - 2; t <= right + 2; t++) {
        if (0 <= t && t <= rght) {
            long ans = rs;
            ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
            max = Math.min(max, ans);
        }
    }
    return left;
}

public static double func_c0cfdb88809544b3b437b1852816f302(int r, int
s, int n, int p, int q){
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
```

```
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    return ret;
}

public static long func_fefe730f2b5b493783d37ef6cb6758eb(Scanner in){
    long B = in.nextLong();
    int n = in.nextInt();
    long[] A = new long[37];
    Arrays.fill(A, 0);
    return B;
}

public static long func_3c1932c455174b109b6c0d8e686f7cb2(long x, long
best, long sum, Long down){
    long now = Math.max(down, Math.max(x - down, sum - x));
    best = Math.min(best, now);
    return now;
}

public static int func_0be3559375314dbe9c965fca6fcbbdff(Scanner input)
{
    int n = input.nextInt();
    int p = input.nextInt(), q = input.nextInt(), r = input.nextInt(),
s = input.nextInt();
    return p;
}

public static long func_4186235a00a34cbdaefdc5ba4ea2950b(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return sum;
}
```

```java
public static int[] func_8573f8fcaf004e30bcf1cffa8ef0b3d3(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    return data;
}

public static long[] func_c2049e7cd58b41ea940ea3ac23fae37b(int r, int
s, int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    return a;
}

public static double func_53d5874a12c34a779da2e8522286c8c9(int w, int
G, double tS, double[] h, double[] S){
    for (int i = 0; i < w; i++) {
        S[i] = (h[i] + h[i + 1]) / 2;
        tS += S[i];
    }
    double[] ans = new double[G - 1];
    return tS;
}

public static double func_a42ac0f97bbd43d8aae5ea09220055a8(int i,
int[] y, int[] x, double xx, double x1){
    double x2 = x[i + 1];
    double y1 = y[i];
    double y2 = y[i + 1];
    return y1 + (y2 - y1) * (xx - x1) / (x2 - x1);
}

public static long func_b74c875f8791449a907c02c3365334c8(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
```

```java
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    return LO;
}

public static double func_193857f3c1e24ce181cb0b5541b719dd(double t,
double k, double res, double d1, double dx){
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return res;
}

public static long[] func_6476235e0b21474e8d9b907fe2d85861(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    return RS;
}

public static long func_e660047c22fc4ac3ad5bb1dcb7c21784(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    return minimal;
}

public static long func_db60cc68501a476ea9a3a74b5b789760(int n, int s,
int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
```

```
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    return left;
}

public static long[] func_c31089dedec947f9a449ec8573305fd9(int cn, int
N, PrintWriter out, Scanner in){
    long p = in.nextInt();
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] d = new long[N];
```

```
    for (int i = 0; i < N; i++) {
        d[i] = (p * i + q) % r + s;
    }
    out.println(String.format("Case #%d: %.11f", cn,
ProblemA.solve(d)));
    return d;
}

public static int func_55d7b17477514066baa0af9f1547c35e(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
```

```
    return x0;
}

public static double func_cb330ea9a87b40858cec1e670f0869b6(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
    }
    return temp;
}

public static long[] func_6d775717aaff4fedb175d6f129203e1f(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return dp;
}

public static long func_4bc886b86f494f8db3da16ae7f3e46c7(int lo, int
```

```
i, int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    return diff1;
}

public static double func_295d5b8de9f44245aac605696598e2d0(double
curS, double S, double xf, double ts, ArrayList<Double> ans){
    ans.add(xf);
    ts -= S - curS;
    curS = 0;
    return curS;
}

public static long func_b7dff9effaeb4658a7f885d7b4bd1668(long x, long
best, long sum, Long down){
    long now = Math.max(down, Math.max(x - down, sum - x));
    best = Math.min(best, now);
    return now;
}

public static double func_7f61449a14404f4ba9bbe76a3aa0eca9(double hi,
double lo, double b, double a, double used){
    double mi = (hi + lo) / 2.0;
    double cand = ((b - a) * used + a + (b - a) * mi + a) * (mi -
used) / 2.0;
    return cand;
}

public static double func_09634aaa5e374fae8dce622d09ba925e(int ulidx,
```

```
int llidx, double xe, double x1, double d1){
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    d2 = A.l[llidx + 1].x - A.l[llidx].x;
    d3 = xe - A.l[llidx].x;
    double ysl = A.l[llidx].y + (d1 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    double yel = A.l[llidx].y + (d3 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    return d2;
}

public static long[] func_90ddff0739a342eb80497bbe33066f49(int N){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    return arr;
}

public static double[] func_699a84505fb34d0a9b232da462ae87fe(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    return len;
}

public static int[] func_8a268b4151fe4b838ad7a44bf5151cb1(int s, int
```

```
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    return array;
}

public static long func_751209d7e9054188a02230289e854256(int i, int n,
long[] a){
    long l = a[i] - 1;
    long r = a[n - 1] - 1;
    return l;
}

public static long func_a5b1bcd45d59442f870879096eecf5ee(int rght,
long[] pref){
    long sum = pref[rght];
    int left = -1, right = rght;
    return sum;
}

public static long[] func_8ae9d6fd7db34b45bcff1784ab1d21bd(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    for (int i = 0; i < n; i++) {
        int low = i;
        int high = n - 1;
        long prefixSum = (i == 0 ? 0 : rsum[i - 1]);
        while (high - low > 1) {
            int mid = (high + low) / 2;
            long sum = (rsum[mid] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            if (sum <= suffSum) {
                low = mid;
            } else {
                high = mid;
            }
```

```
        }
        for (int j = low; j <= high; j++) {
            long sum = (rsum[j] - prefixSum);
            long suffSum = rsum[n - 1] - sum - prefixSum;
            long max = Math.max(sum, suffSum);
            max = Math.max(max, prefixSum);
            best = Math.max(best, rsum[n - 1] - max);
        }
    }
    return rsum;
}

public static long func_9a75d85ac83146eea9dc324f5d2f6b26(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    return lo;
}

public static double func_14f653ef46e24c078f6c0724d0f1ffdf(double
guess, double targetArea, double low, double curans, double high){
    double a = A.area(curans, guess);
    if (a < targetArea) {
        low = guess;
    } else {
        high = guess;
    }
    return low;
}

public static long[] func_3c0a56ed10ea4a5691c12c48a9e41527(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
```

```
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return nums;
}

public static ArrayList func_5b7fb619c348443abf7093ea898f3143(double
y0, double total, double x0, ArrayList<Double> lx, ArrayList<Double>
ly){
    for (int i = 0; i < lx.size(); ++i) {
        double xx = lx.get(i);
        double yy = ly.get(i);
        total += (x0 + xx) * (y0 - yy);
        x0 = xx;
        y0 = yy;
    }
    return ly;
}

public static double func_bccd29251a3a435d9072e10ce1745122(double
right, double dy1, double left, double dy2, double dx){
    double mid = (left + right) / 2.0;
    double dy2_ = (dy2 - dy1) * (mid / dx) + dy1;
    double S2 = (dy1 + dy2_) * mid / 2.0;
    return S2;
}

public static long func_f9c0bf34193a4f8ca326952a8f2ebd9b(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
```

```
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return sum;
}

public static int func_f47c55a0c254471e9862b6420164dc86(int lo, int i,
int hi, long tot, long[] S){
    int mid = (lo + hi) / 2;
    long center = S[i + 1] - S[mid];
    long left = tot - center;
    return mid;
}

public static long[] func_a426e1cdcc2e43dda0d8f44b6945589d(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    double res = (A[N] - ans) * 1.0 / A[N];
```

```
    return A;
}

public static double func_56fb4c7ab2ce4ba49cb63c2cdcbb9506(int ulidx,
int llidx, double d3, double x1, double d1){
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    return ysu;
}

public static String func_9fb3043946fd4d609739f79f37101f96(int[] sol,
int[] prob, int[] len){
    for (int i = 0; i < sol.length; i++) {
        sol[i] = i;
    }
    for (int a = 0; a < sol.length; a++) for (int b = a + 1; b <
sol.length; b++) {
        if (prob[b] > prob[a]) {
            Die.swap(len, a, b);
            Die.swap(prob, a, b);
            Die.swap(sol, a, b);
        } else if (prob[b] < prob[a]) {
            continue;
        } else {
            // probs same
            continue;
        }
    }
    boolean swapped;
    do {
        swapped = false;
        for (int a = 0; a < sol.length - 1; a++) {
            if (prob[a] == prob[a + 1]) {
                if (sol[a + 1] < sol[a]) {
                    Die.swap(sol, a, a + 1);
                    swapped = true;
                }
            }
        }
    } while (swapped);
    return Die.tostr(sol);
}

public static boolean func_d3ae5c857a684997b8cd15c8e93a2485(int n,
long l, long r, long[] a){
    long m = (l + r) / 2;
```

```
    boolean ok = true;
    for (int i = 0; i < n; i++) {
        if (a[i] > m)
            ok = false;
    }
    return ok;
}

public static int func_1321702f19d44abbaa70230f85e84c1b(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return j;
}

public static long func_70ccff3201ab45ebaa12ec50991d4b45(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return sum;
}

public static int[] func_d2f0ee67bfe84a0589e3ef7350300188(int n, int
r, int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
```

```
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return values;
}

public static long func_f38e9f92804244a2a7b88e43594dee1b(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    if (v1 < v2) {
        min = med1;
    } else {
        max = med2;
    }
    return v1;
}

public static double func_921b8c552e1a4fa3a182c3153b0f61c3(int ulidx,
int llidx, double xe, double x1){
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    d2 = A.l[llidx + 1].x - A.l[llidx].x;
    d3 = xe - A.l[llidx].x;
    double ysl = A.l[llidx].y + (d1 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    double yel = A.l[llidx].y + (d3 / d2) * (A.l[llidx + 1].y -
```

```
A.l[llidx].y);
    return yeu;
}

public static double func_192c076bc3734f019ab885a1dd4cbe50(int keep,
long win, long total){
    double payoff = 36.0 * win / keep;
    double ret = payoff - total;
    return payoff;
}

public static long[] func_c65c04d6f013476db3f36476166eec4e(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    return pref;
}

public static long func_7ee551904aa548209326319e454c5938(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
```

```
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return min;
}

public static long func_2f70c93ab48946f0b071b11ae8f13375(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    return t2;
}

public static double func_2018eb26b6b34551924924ba14ca806a(double k,
double d1, double want, double dx){
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return t;
}

public static long func_90d8b0697efe41a09f7bf3989cf9716e(int c, long
B, long[] left){
    long[] a = left.clone();
    long tmpB = B;
    for (int i = 0; i < c; i++) {
        tmpB -= a[c - 1] - a[i];
        a[i] = a[c - 1];
    }
    for (int i = c; i < A.C; i++) {
        if (a[i] == a[0]) {
            a[i]++;
            tmpB--;
        }
```

```
    }
    return tmpB;
}

public static long func_26f696d7d63c4e008a91fd89b9d32b89(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return temp;
}

public static long func_aa1e94533c0d488f90b4a4b7a8a5a364(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
```

```
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return sum;
}

public static long[] func_f394fba1bb104c629c694007ae2240e7(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return arr;
}

public static double func_7dc631b73641407886a1089c6d155e59(double dg,
double lg, double p){
    int counter = 0;
    while (lg + 0.00000001 < dg && counter < 1000) {
        counter++;
        double s = (lg + dg) / 2;
        double tp = C.povrsinaDo(s);
        if (tp < p)
            lg = s;
```

```
        else
            dg = s;
    }
    if (counter == 1000)
        System.out.println((dg - lg) + " " + lg + " " + dg);
    return dg;
}

public static int func_4b78f62064e14d908e1dd8a3c0e0a063(int lo, int i,
int hi, long tot, long[] S){
    int mid = (lo + hi) / 2;
    long center = S[i + 1] - S[mid];
    long left = tot - center;
    return mid;
}

public static long[] func_214909e4fd334f9a9748f4eb09883b02(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    return cum;
}

public static long func_a95869af1bee4d8295c0ad16fd97c7ae(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    return t2;
}

public static long[] func_86505ce73b6442148cba0eb76182f4e4(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
```

```
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    return a;
}

public static long[] func_090afca588734afbb98882d5a3eac10e(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    return sum;
}

public static long func_566c47bbc00f44efa8cc44ebb786cf89(int N, long[]
is){
    N = 37;
    Arrays.sort(is);
    long A = 0;
    // for(long i:is)B += i;
    long K = 0;
    return K;
}

public static long[] func_662f108a55f945c4afcaa8a8c74ea70a(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return LS;
}

public static int[] func_e68f10e442164591a58b658e437f8f64(int r, int
count, int q, int s, long p){
```

```
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    return qty;
}

public static double func_1f3ca36b37264151ac7df50c553ed6f2(int U, int
W, int G, int L, double total){
    for (int i = 0; i < L - 1; i++) {
        total -= (IrregularCakes.lower[i + 1].getY() +
IrregularCakes.lower[i].getY()) * (IrregularCakes.lower[i + 1].getX()
- IrregularCakes.lower[i].getX()) / 2;
    }
    for (int i = 0; i < U - 1; i++) {
        total += (IrregularCakes.upper[i + 1].getY() +
IrregularCakes.upper[i].getY()) * (IrregularCakes.upper[i + 1].getX()
- IrregularCakes.upper[i].getX()) / 2;
    }
    IrregularCakes.out.println("Case #" + (++IrregularCakes.caseNo) +
": ");
    for (int g = 1; g < G; g++) {
        double part = g * total / G;
        double lb = 0;
        double ub = W;
        while (ub - lb > 0.000000001) {
            double next = (ub + lb) / 2;
            double area = IrregularCakes.area(next);
            if (area > part) {
                ub = next;
            } else {
                lb = next;
            }
        }
        IrregularCakes.out.println(lb);
    }
    return total;
}

public static double[] func_c6aa4a6da50b4d8a831a963c197ec339(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
```

```
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    return low;
}

public static long func_8f0b107168ff48659de11aad2ff53fee(int N, int[]
arr, long sum, long[] LS){
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return HI;
}

public static long[] func_da1db89cd7e749f8bac8436275c59c9e(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
```

```
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    return amt;
}

public static long func_4c1e4848357b46d2a29a1634ad48b840(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return min;
}

public static int func_54d0d675051a4fc4a9e15493405c6000(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
```

```
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    return y0;
}

public static boolean func_b4852e437b2948e9a61258e4018018eb(int
joined, long lowestBet, long required, long budget, long[] amount){
    for (int i = 0; i < 37; i++) {
        required += Math.max(0, i < joined ? (lowestBet - amount[i]) :
(lowestBet + 1 - amount[i]));
    }
    return required <= budget;
}

public static int[] func_05b535b37e784e31aebc66890a6ea044(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    return data;
}

public static int func_6acbc63974694bc7ae15a28ebebb8f6a(int l, int px,
int[][] pos, double[] bound){
    int py = pos[0][1];
    for (int i = 1; i < l; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] -= (ny - py) / div * (j - px) + py;
        }
```

```
        px = nx;
        py = ny;
    }
    px = pos[l][0];
    return px;
}

public static long[] func_8de944ec301f4891a4835bfa6cee97a9(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    return pref;
}

public static int[] func_386a6111b0d34b15823994c3791f096a(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    return data;
}

public static long[] func_f5d1cf61be3342929cac300dda5767dc(int p, int
q, int n, Scanner scanner){
    int r = scanner.nextInt();
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    return partials;
}
```

```
public static int func_428cb4019c984419a1cd548b7ec02737(int lo, int i,
int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    return lo;
}

public static long func_6cc755e1ffba4a07b1de2b927d1eb431(int lo, int
N, int i, long left, long[] psum){
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    if (lo > i + 1) {
        t = psum[lo - 1] - psum[i];
        a = Math.min(a, Math.max(t, left - t));
    }
    long get = psum[N] - Math.max(psum[i], a);
    return t;
}

public static long[] func_7762f353cbe9428abacda346822d88e7(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
```

```
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return A;
}

public static long func_b008c4e4e7dc455ba62f9ec1f4f90060(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return temp;
}

public static long func_44423c962ee0437f93c1c68e437ab177(int n, int N,
long k, long Y, long[] is){
    for (int i = n; i < N; i++) Y += Math.max(0, k + 1 - is[i]);
    return Y;
}
```

```
public static int func_242f9ad9d6b643ee8e709c9afab6e55f(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return first;
}

public static long func_073a7b4143ba4cb08d3c9a3278a52594(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
```

```
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return min;
}

public static double func_77323cf0f0004ec289078daf25a3e932(int W, int
U, int G, int L, Scanner input){
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    return firstArea;
}

public static double[] func_9855400dfbee470095249b0bebecb167(double
min, double max){
    double mid = (min + max) / 2;
    double[] poly = new double[8];
    poly[0] = mid;
    poly[1] = 1001;
    poly[2] = mid;
    poly[3] = -1001;
    return poly;
}

public static int func_0d97350191a6425dae0f00ad63cb105d(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
```

```
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return beforeHalf;
}

public static int[] func_39b7fa21e8f54640b1acee646b7a650c(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    return a;
}

public static double func_2e334e9668e44fe699b52db0819fa3ac(int i, int
G, double high, double all, double low){
    double T = all / G * (i + 1);
    while (high - low > 1E-9) {
        double mid = (high + low) / 2;
        if (A.S(mid) < T) {
            low = mid;
        } else {
```

```
            high = mid;
        }
    }
    return T;
}

public static int[] func_00c81afeeee44fb7a648bb748947905c(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    return a;
}

public static double[] func_a4a6c06c37354b66a3d40a5a7e2bff40(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
```

```
    double[] low = new double[w + 1];
    return low;
}

public static long[] func_fa0a2b893b2e470fbae9c651f82fdcb6(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    return a;
}

public static long func_f665d69bcc62448b898273e4ba1c8bcb(int z, int n,
int index, long min){
    while (z < n - 1 && Math.abs(A.sum(z, index) - A.sum(index + 1, n
- 1)) > Math.abs(A.sum(z, index + 1) - A.sum(index + 2, n - 1))) {
        index++;
    }
    min = Math.min(min, Math.max(A.sum(0, z - 1), Math.max(A.sum(z,
index), A.sum(index + 1, n - 1))));
    return min;
}

public static long func_691176da97f342fdaf41e39d3ff31701(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
```

```
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return L;
}

public static double[] func_231831b9fd654914aca7721ee8a03930(int y0,
int w, int x0, int u, Scanner in){
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    return upper;
}

public static long func_17b842eb53c943a1bf32d9be482556b4(int N, int j,
int i, long sum, long[] dp){
    long a = dp[i + 1];
    long b = dp[j] - dp[i + 1];
    long c = dp[N] - dp[j];
    long left = sum - Math.max(a, Math.max(b, c));
    return b;
}

public static int[] func_5ef15c399d8b4bf390992a661ead31eb(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    double ans = 0;
    return a;
}

public static double func_4126346e4dbe47f893281d805d4d6fe1(int ulidx,
```

```
int llidx, double rtn, double x1, double x2){
    // System.out.println(x1 + " " + ulidx + " " + llidx);
    double xe = Math.min(Math.min(A.u[ulidx + 1].x, A.l[llidx + 1].x),
x2);
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    d2 = A.l[llidx + 1].x - A.l[llidx].x;
    d3 = xe - A.l[llidx].x;
    double ysl = A.l[llidx].y + (d1 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    double yel = A.l[llidx].y + (d3 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    rtn += (xe - x1) * ((ysu - ysl) + (yeu - yel)) / 2;
    if (A.u[ulidx + 1].x < A.l[llidx + 1].x) {
        x1 = A.u[ulidx + 1].x;
        ulidx++;
    } else {
        x1 = A.l[llidx + 1].x;
        llidx++;
    }
    return yeu;
}

public static int func_458ea50673f1440280d653ff01bbac00(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
```

```
    return x;
}

public static double func_eafbe7f5d2f044349968607f361b0652(int r, int
p, int q, int n, Scanner scanner){
    int s = scanner.nextInt();
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    long total = partials[n];
    return 1 - (double) (best) / total;
}

public static long[] func_2a8c180258da479d8c34008148fe519c(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    long[] pref = new long[n];
    return ar;
}

public static long func_a4b09734ae74413cbad0921bd4291ab6(int numtie,
int i, long waste, long initadd, long[] v){
    if (v[i] == v[i - 1]) {
        v[i]++;
        initadd++;
        waste += numtie;
    }
    long bet = i * (v[i] - 1);
    long min = 0;
    for (int j = 0; j < i; j++) {
        min += v[j];
    }
    return bet;
}
```

```
public static double func_72adcd8cb4474423871e850215c60921(double
target, double hi, double lastx, double lo){
    double mid = 0;
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    return mid;
}

public static long func_397653c20fb54c07ae62a18e53cd6373(int
beforeHalf, int n, long sum, long beforeHalfSum, long[] A){
    while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum / 2)
{
        beforeHalfSum += A[beforeHalf];
        beforeHalf++;
    }
    long max = sum;
    if (beforeHalf > 0) {
        max = Math.max(beforeHalfSum, sum - beforeHalfSum);
    }
    if (beforeHalf <= n - 1) {
        long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
        max = Math.min(max, value);
    }
    return beforeHalfSum;
}

public static int[] func_fdba07e8d3894bfabb96bc396ec29e6d(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    double ans = 0;
    return a;
}

public static ArrayList func_d34af4a862974ab6acc037fd71b9ff9a(int i,
double[] x, double[] y, ArrayList<Double> lx, ArrayList<Double> ly){
```

```
    lx.add(x[i]);
    ly.add(y[i]);
    return ly;
}

public static long[] func_690f12ce0626454d8899cd3dc5f98952(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return A;
}

public static int func_b9aa1632b92c404a8030d4dffd21e889(int Lb, int
La, int Fa, int Fb){
    int Tab = 100 * La + Lb * (100 - Fa);
    int Tba = 100 * Lb + La * (100 - Fb);
    return Tba;
}
```

```java
public static int func_5dc7fa4264a24b0e924f04fd269e0917(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return hi;
}

public static long func_1d26f5b7d2964ff28b968ef21eb769f4(int
winningThings, long payMoney, long low, long budget, long[] x){
    for (int i = 0; i < winningThings; i++) {
        payMoney += low - x[i];
    }
    if (payMoney > budget) {
        return -1;
    }
    --low;
    long high = budget + 1;
    while (high - low > 1) {
        long mid = low + high >> 1;
        if (A.canDo(budget, x, winningThings, mid)) {
            low = mid;
        } else {
            high = mid;
        }
    }
    return high;
}

public static double func_5ce139ff27aa4c819088b7a855d383a6(int l,
double bx, double ans, double ax){
    double ay = A.highy[l];
    double by = A.highy[l + 1];
    if (ax > A.highx[l]) {
        ay = A.highy[l] + (ax - A.highx[l]) / (A.highx[l + 1] -
A.highx[l]) * (A.highy[l + 1] - A.highy[l]);
    }
    if (bx < A.highx[l + 1]) {
        by = A.highy[l + 1] + (bx - A.highx[l + 1]) / (A.highx[l] -
A.highx[l + 1]) * (A.highy[l] - A.highy[l + 1]);
    }
    ans -= (bx - ax) * (1000 - (ay + by) / 2);
    return by;
}

public static long func_53db802d088849ea9f55f823b074105f(int s, int r,
int p, int q, int n){
```

```
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    return best;
}

public static long func_0e1d1234e07140abb4a666785afb7eb5(long sum){
    long HI = sum;
    int pow2 = 1;
    return HI;
}

public static int func_b95c7a786d184d1f8dd609a8b214899e(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return second;
}

public static int[] func_668539a721d34ef79bd48b6870b2f40f(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    return data;
}

public static long func_1fccf08de3e74d0db3718a62464e86dd(int j, int i,
long win, long required, long toAdd){
    required += toAdd * j;
    win += toAdd * i;
```

```
    return required;
}

public static double func_76e9f1bd85c94a3a9a39890831858bc2(int n, int
r, int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return (double) res / sum;
}

public static long func_df4f102a964d4656abf8148058d8b003(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
```

```
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return HI;
}

public static double[] func_4db4fd21400a45b983f26b039db15e8c(int g,
int ttt, int w, double trazenaPovrsina, PrintWriter out){
    double[] res = new double[g - 1];
    for (int i = 0; i < g - 1; i++) {
        double p = (i + 1) * trazenaPovrsina;
        double lg = 0;
        double dg = w;
        int counter = 0;
        while (lg + 0.00000001 < dg && counter < 1000) {
            counter++;
            double s = (lg + dg) / 2;
            double tp = C.povrsinaDo(s);
            if (tp < p)
                lg = s;
            else
                dg = s;
        }
        if (counter == 1000)
            System.out.println((dg - lg) + " " + lg + " " + dg);
        res[i] = (lg + dg) / 2;
    }
    out.printf("Case #%d: ", ttt);
    out.println();
    return res;
}

public static double func_69b4ac7bf8af4e38af5fef002226cdce(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    lastx = mid;
    return mid;
}

public static long func_964c187bb974491fbd63a04c09543fa2(int N, long
r, long p, long s, long q){
```

```
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    return L;
}

public static long[] func_54e967b18d5546228e974167ad4e1e9f(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    return cum;
}

public static long func_4e25b97c275446a19d8822c72fda475f(int lo, int
i, int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    return diff1;
```

```
}

public static double[] func_160509c227054150b3cf723c98456b08(int w,
int[] lx, int[] ly, int[] ux, int[] uy){
    double[] h = new double[w + 1];
    for (int i = 0, pu = 0, pl = 0; i <= w; i++) {
        double x = i;
        if (i > ux[pu + 1])
            pu++;
        if (i > lx[pl + 1])
            pl++;
        double lH = (ly[pl] * (lx[pl + 1] - x) + ly[pl + 1] * (x -
lx[pl])) / (lx[pl + 1] - lx[pl]);
        double uH = (uy[pu] * (ux[pu + 1] - x) + uy[pu + 1] * (x -
ux[pu])) / (ux[pu + 1] - ux[pu]);
        h[i] = uH - lH;
    }
    double[] S = new double[w];
    double tS = 0;
    return h;
}

public static long func_8e27ab82438d4ca1a2efe027e20537ce(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
```

```
    return sum;
}

public static double func_642b98042b7444f9a132e210c067f83c(int ulidx,
double xe, double x1){
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    return yeu;
}

public static long func_a398dd2a3ce94f96ba8287f4862a8f6e(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    return temp;
}

public static long func_30f438f45d6d465ab06f03f449d20697(int numtie,
int i, long bet, long waste, long dif){
    waste += numtie * dif;
    bet += i * dif;
    numtie++;
    return waste;
}

public static int func_a5b1d7632a0f4fd482028bc9694b5f0c(int n, int
rght, long max, long[] pref){
```

```
    long rs = pref[n - 1] - pref[rght];
    long sum = pref[rght];
    int left = -1, right = rght;
    while (left < right - 1) {
        int mid = (left + right) >> 1;
        if (pref[mid] * 2 >= sum) {
            right = mid;
        } else {
            left = mid;
        }
    }
    for (int t = right - 2; t <= right + 2; t++) {
        if (0 <= t && t <= rght) {
            long ans = rs;
            ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
            max = Math.min(max, ans);
        }
    }
    return left;
}

public static double func_0e1a64da1bbe43dba637b98c4d7d012f(int g,
double hi, double cur, double used){
    cur = 0.0;
    used = hi;
    --g;
    return cur;
}

public static long func_ce5404841a914803abd857e6adb942bd(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    return sum;
}

public static long[] func_b6d11191fbea4a28bee7c4dc791cbbff(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
```

```
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return a;
}

public static long func_4072b091a32845aaabfd395a70b1ee79(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return sj;
}
```

```java
public static long func_41d3e0d9cd704810abc69f4cc6093ec9(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return temp;
}

public static long func_97c5f9dd30d84303b67f6bf2313da8a8(Scanner in){
    int N = in.nextInt();
    long p = in.nextInt();
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    return s;
}

public static double func_b1ccca7d8c2648218f36a4d952d620bc(double low,
double curans, double high, PrintWriter out){
    curans = (low + high) / 2;
    out.println(curans);
    return curans;
}

public static long[] func_846bc3571f3d4c7f8f5f9944ac66f131(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    long[] pref = new long[n];
    for (int i = 0; i < n; i++) {
        pref[i] = ar[i];
        if (i > 0) {
            pref[i] += pref[i - 1];
        }
    }
    return ar;
}
```

```
public static int[] func_8205505b58904fc7835850a2baf5f819(int cas,
Scanner in){
    int n = in.nextInt();
    int[] L = new int[n];
    final int[] P = new int[n];
    Integer[] I = new Integer[n];
    for (int i = 0; i < n; i++) L[i] = in.nextInt();
    for (int i = 0; i < n; i++) {
        P[i] = in.nextInt();
        I[i] = i;
    }
    Arrays.sort(I, new Comparator<Integer>() {

        public int compare(Integer A, Integer B) {
            if (P[A] != P[B])
                return P[B] - P[A];
            return A - B;
        }
    });
    System.out.printf("Case #%d:", cas);
    for (Integer x : I) System.out.print(" " + x);
    System.out.println();
    return P;
}

public static long func_6eff044ab48749479524994e5e005f9a(int q, int s,
int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    return x;
}

public static long[] func_292c018702db43cca796233f2b3ca548(Scanner br)
{
    int n = br.nextInt();
    long p = br.nextInt();
    long q = br.nextInt();
    long r = br.nextInt();
    long s = br.nextInt();
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
```

```
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return rsum;
}

public static long func_3b350cd1b21d487d9e9d0e7ad066de8e(int j, int i,
long third, long ans, long[] A){
    long first = A[j + 1];
    long second = A[i + 1] - A[j + 1];
    ans = Math.min(ans, Math.max(first, Math.max(second, third)));
    return first;
}

public static int[] func_b0b2d142983442cfb02e08c6fd3ba8b8(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    return array;
}

public static long func_575ddd18d05a4ca9aa20f360e43adf28(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
```

```
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    return sum;
}

public static long func_cffd081210624dd5b473569256b35375(int
winningThings, long budget, long[] x){
    long low = x[winningThings - 1];
    long payMoney = 0;
    for (int i = 0; i < winningThings; i++) {
        payMoney += low - x[i];
    }
    if (payMoney > budget) {
        return -1;
    }
    --low;
    return low;
}

public static long[] func_52e2ac03e8e04c97a2a2301c2422cb50(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return dp;
```

```
}

public static int func_b50469877d0348369b71356d6c3f5b56(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return second;
}

public static long func_0ce930fe7820447dabac7e9c115d9e63(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    return hi;
}

public static PrintWriter func_9248edc379454c9ea1c9c533f56b89ea(int i,
int n, double[] exp, PrintWriter pw){
    System.out.print("Case #" + (i + 1) + ": ");
```

```
    for (int j = 0; j < n; j++) {
        int pos = 0;
        for (int k = 1; k < n; k++) {
            if (exp[k] > exp[pos]) {
                pos = k;
            }
        }
        pw.print(pos + " ");
        System.out.print(pos + " ");
        exp[pos] = -1;
    }
    pw.println();
    System.out.println();
    return pw;
}

public static int func_e9b719bd6e214dbbb0e3fc130c5cce93(int pN){
    Scanner sc = new Scanner(System.in);
    final int caseN = sc.nextInt();
    final A[] solvers = new A[caseN];
    StringWriter[] outs = new StringWriter[caseN];
    for (int i = 0; i < caseN; i++) {
        solvers[i] = new A();
        outs[i] = new StringWriter();
        solvers[i].out = new PrintWriter(outs[i]);
        solvers[i].out.printf("Case #%d: ", i + 1);
        solvers[i].read(sc);
    }
    Thread[] ts = new Thread[pN];
    return caseN;
}

public static double func_d5c4dbd3ef704cf5bb493c45a0d02021(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    double ans = 0;
    return ans;
}

public static double func_c5878de7912f4f25a336814c21c0a460(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
```

```
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return ans;
}

public static double func_a44734bf079a49e1951a923a750923e8(int ulidx,
int llidx, double x1, double x2){
    // System.out.println(x1 + " " + ulidx + " " + llidx);
    double xe = Math.min(Math.min(A.u[ulidx + 1].x, A.l[llidx + 1].x),
x2);
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    d2 = A.l[llidx + 1].x - A.l[llidx].x;
    d3 = xe - A.l[llidx].x;
    double ysl = A.l[llidx].y + (d1 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    double yel = A.l[llidx].y + (d3 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    return yeu;
}

public static double func_65051d7cb43a4b58963224fdbd34260b(double cur,
double used, double b, double a, double req){
```

```
    double hi = 1.0;
    double lo = used;
    for (int j = 0; j < 100; ++j) {
        double mi = (hi + lo) / 2.0;
        double cand = ((b - a) * used + a + (b - a) * mi + a) * (mi -
used) / 2.0;
        if (cur + cand >= req) {
            hi = mi;
        } else {
            lo = mi;
        }
    }
    return hi;
}

public static long[] func_6c1524ad26014856996a636a149e3353(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    return A;
}

public static int func_bcb20ae3eece49ff807593deb0ce13b4(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
```

```
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    return x0;
}

public static double func_cba6464174ee4ed19b202f281e891ce2(double y0,
double total, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    for (int i = 0; i < lx.size(); ++i) {
        double xx = lx.get(i);
        double yy = ly.get(i);
        total += (x0 + xx) * (y0 - yy);
        x0 = xx;
        y0 = yy;
    }
    return Math.abs(total) / 2;
}

public static long func_e59ae4c583ea4a25be5287f686d33d8e(long
minspaces, long B, A.MaxEntry e){
    long budget = B;
    budget -= minspaces + e.fix;
    return budget;
}

public static double[] func_480125da189645a89d7065d496d9f38f(int n,
double[] y2, double[] y1){
    /*                   for (int i = 0; i < n; i++) out.print(x[i] +
" ");
    out.println();
    for (int i = 0; i < n; i++) out.print(y1[i] + " ");
    out.println();
    for (int i = 0; i < n; i++) out.print(y2[i] + " ");
    out.println();*/
    double[] yy = new double[n];
    for (int i = 0; i < n; i++) yy[i] = y2[i] - y1[i];
```

```
    return yy;
}

public static long func_918448783114420685f008c7df259636(int numtie,
int i, long bet, long waste, long[] v){
    long dif = v[i + numtie] - v[i];
    waste += numtie * dif;
    bet += i * dif;
    return dif;
}

public static long func_653064334e1d420ca39986f654d2935f(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return lo;
}

public static int func_c52b650add5d41ca84e054149af0892b(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    return x;
}

public static long func_882a4938e20f41979fbceb353057ae7a(int N, int
beforeHalf, long[] best, long[] A){
    long beforeHalfSum = 0;
```

```
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return sum;
}

public static long[] func_fbc77f778ef24d3681027025481e129b(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
```

```
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return A;
}

public static long func_6d92a77f99f543c1913b1a5b9abffca3(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return beforeHalfSum;
}

public static int[] func_052bc0f4622e4d3d9024bf82c24fbe33(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    return a;
}

public static int func_5f96d232ea644d49a5a23e947ea0f0e7(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
```

```
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return beforeHalf;
}

public static long func_a11273137c8843c6a569b5b897d1963b(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
```

```
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return l;
}

public static int[] func_224f802f25e948d6b7bcfc4eaf0652db(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return arr;
}

public static int func_d904425d8a60472487b4b2e192ae31c7(int q, int s,
int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    return index;
}

public static double func_444850f237584394b2032d9c32412a01(int p2,
int[] x2, int[] y2){
```

```java
    double a2 = y2[p2 + 1] - y2[p2];
    double b2 = x2[p2] - x2[p2 + 1];
    double c2 = a2 * x2[p2] + b2 * y2[p2];
    return a2;
}

public static long[] func_f03641fdc05048589576787533b70efe(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    return a;
}

public static int func_7742655ad0474146bc2724e6e34935f6(int i, int x,
int y, long tmp, long sum){
    if (tmp * 3 <= sum) {
        x = i;
    }
    if (tmp * 3 <= sum * 2) {
        y = i;
    }
    return y;
}

public static int func_6688f71314814a2fbd8bc40071589925(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    return beforeHalf;
}

public static double func_9d2ee4bd14ac41af8b6728ef08ead755(int o,
int[] v, double[] l, double[] p){
    int j = 0;
    double best = 1e200;
    double lsum = 0, rsum = 0;
    double psum = 0;
    double m = 0;
    double q = 1;
    for (int i = 0; i < o; i++) {
        m += l[v[i]];
        psum += p[v[i]] * q;
        rsum += p[v[i]] * q * m;
```

```
        q *= 1 - p[v[i]];
    }
    m = 0;
    q = 1;
    return m;
}

public static long func_6c7e288ff0d7433c8189d60253a370a7(int N,
Scanner in){
    long p = in.nextInt();
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return q;
}

public static double func_67fa8e07c0c04861bf65bef5e0227c13(double l,
double s, double r, double mid, double curArea){
```

```
    if (s > curArea) {
        r = mid;
    } else {
        l = mid;
    }
    mid = (l + r) * .5;
    return mid;
}

public static int func_b5cc5389f0c54ef6a72cf9b98d0c7851(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return index;
}

public static double func_7d5ad95c52e644588c80bcb7ec8746ed(double a1,
double cx, double c1, double nx, double b1){
    double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
    cy1 = (c1 - a1 * cx) / b1;
    my1 = (c1 - a1 * mx) / b1;
    return mx;
}

public static long func_3796e5bff9e94f9bb9617e603a291291(int
winningThings, long high, long low, long budget, long[] x){
```

```
    while (high - low > 1) {
        long mid = low + high >> 1;
        if (A.canDo(budget, x, winningThings, mid)) {
            low = mid;
        } else {
            high = mid;
        }
    }
    return high;
}

public static long func_43400ecf1a96457287a166df1bf8e840(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return rest;
}

public static long func_9279a6dc16ed49afa9ce0e71caee0ee3(int t, long
rs, long sum, long max, long[] pref){
    long ans = rs;
    ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
    max = Math.min(max, ans);
    return max;
}

public static long func_19828780e2bd474dbbd128c702ad498f(int r, int p,
int q, long curr){
    curr *= p;
    curr += q;
    curr %= r;
    return curr;
}

public static double func_d575e3ba590a403090da0794e837d342(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    return k;
```

```
}

public static int func_fbafb52f2d7a42cb8ac77984b52e7263(int lo, int
hi, int i, long left, long[] psum){
    int mid = (lo + hi) >> 1;
    long s1 = psum[mid] - psum[i];
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return mid;
}

public static double func_730ea1e4f33944c8b5ab3fec45288497(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    return d;
}

public static long func_36a649430dd94e0ca855ea33c7325043(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    return res;
}

public static int[] func_2b5214b0677346ef8dd524da21541fdf(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    return arr;
}

public static double func_6c17a9eda7a448a9b910ac5e4e1acb70(Scanner
```

```
input){
    int n = input.nextInt();
    int p = input.nextInt(), q = input.nextInt(), r = input.nextInt(),
s = input.nextInt();
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return res;
}

public static int[] func_154c67f075264138a893f9dab0f715fd(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    return a;
}

public static long func_9cdbc3d115ab454ebd9d0caffeace1d(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
```

```
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return temp;
}

public static long func_b1266109e8fd480696e5f9e5fb345105(long mid,
long min, long budget, long max, long[] bets){
    long need = 0;
    for (int i = 0; i < bets.length; i++) {
        if (bets[i] < mid) {
            need += mid - bets[i];
        }
    }
    if (need > budget) {
        max = mid;
    } else {
        min = mid;
    }
    return max;
}

public static long func_169229210c5f48aea8ec10584ef1d7d2(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    return answer;
}

public static long func_1a6d63ef0e1249bfa0d1b2157e353e62(int p, int q,
int n, int r, int s){
    TaskA.a = new long[n];
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    for (int i = 2, j = 1; i < n; i++) {
        while (j + 1 < i && TaskA.sum(0, j + 1) <= TaskA.sum(j + 1,
```

```
i)) {
            ++j;
        }
        minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i, n),
TaskA.sum(j, i)), TaskA.sum(0, j)));
        if (j + 1 < i) {
            minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i,
n), TaskA.sum(j + 1, i)), TaskA.sum(0, j + 1)));
        }
    }
    return minimal;
}

public static int func_827e863ff8754e91bcfa180ed6cc121e(int t, int w,
int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    return y0;
```

```
}

public static int func_3c691eb88a3a4f8ea6239728f051f819(int y0, int
x0, int x, int y, double[] low){
    double k = 1.0 * (y - y0) / (x - x0);
    for (int j = x0 + 1; j <= x; j++) {
        low[j] = k * (j - x0) + y0;
    }
    x0 = x;
    return x0;
}

public static long func_1944730644b944a9aa926af10ef31afe(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return avg;
}

public static double func_050453611de74f3a961c4d1fc1616d20(int i, int
n, long l, long[] a, double max){
    long s = 0;
    long s2 = 0;
    for (int j = 0; j < n; j++) {
        if (j <= i) {
            s += (l - a[j]);
            s2 += (l - a[j]);
        } else if (a[j] < l + 1) {
            s += (l + 1 - a[j]);
        }
    }
    double w = s2 * (36.0 / (i + 1)) - s;
    // System.out.println(i + " " + l + " " + w);
    max = Math.max(max, w);
    return w;
}

public static long func_35a44abb887d4125ac51e070d3cc06be(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
```

```
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return sum;
}

public static int func_d94508d457ea470a837eff558be9592c(int lo, int N,
int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
```

```
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    return lo;
}

public static long[] func_c4926f78c6b947f693980ee4a30ba54d(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return A;
}

public static double func_7f489bda1fde4d5f8b5f7c943ac68864(double y0,
double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    return y0;
}
```

```
public static long func_b7e85ef79d2d4aaeb212756552b31991(int N, int[]
arr, long sum, long[] LS){
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return HI;
}

public static long func_25b7a4f3b875449ab870a0ea50f762e9(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
    return hi;
}

public static long func_b08fa79b51b8457dad538e5a72b915c7(int r, int p,
int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return x;
```

```java
}

public static double func_bd163eb242f44a3e8548026ae1089b8b(double k,
double remain, double x, double y){
    double a = k;
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    double x3 = -1.0 * (d + b) / (2 * a);
    return x3;
}

public static long func_2922ff45e9844cf785d7ab9c41b494f6(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return temp;
}

public static long func_a5b138f7b2d54f718506ca074cb2394c(int right,
int start, int N, int left, long[] sums){
    final int middle = (left + right) >> 1;
    final long midCount = sums[middle] - sums[start];
    final long rightCount = sums[N] - sums[middle];
    if (midCount < rightCount) {
        left = middle + 1;
    } else {
        right = middle - 1;
    }
    return midCount;
}

public static int[] func_cdb3fadf941c4a5da7ac6665c924b025(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
```

```
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return arr;
}

public static int func_fb492723da2a4a20b871066864fa56f5(int t, Scanner
in){
    int w = in.nextInt();
    int l = in.nextInt();
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
```

```
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    return l;
}

public static int func_9728674e00b34d79b3e545af7ae61354(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    if (v1 < v2) {
        min = med1;
    } else {
        max = med2;
    }
    return min;
}

public static int[] func_ccb35b14d6ef4b1e85d4e0bce70a714b(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    for (int i = 0; i < n; i++) {
        while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
        x = Math.min(x, Math.max(sum[i], Math.max(sum[index + 1] -
sum[i], sum[n] - sum[index + 1])));
    // sout(i + " " + index);
    }
    return a;
}

public static int[] func_ba03610f2f6b4a6b8401191e0e465d93(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
```

```
        sum += arr[i];
    }
    long[] LS = new long[N];
    return arr;
}

public static long func_8a014f7cbbdc4abe9b9fbd3b74675f74(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return s;
}

public static int[] func_df687bcd8d54407b8d7524bc699fbcc9(int r, int
p, int q, int s, int n){
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return A;
```

```
}

public static int func_c7578cbec1044d6db4044f6995ac0b53(Scanner sc){
    int w = sc.nextInt();
    int l = sc.nextInt();
    int u = sc.nextInt();
    int g = sc.nextInt();
    double sl = 0;
    return u;
}

public static long func_1b49ea7d3d304eaa90d220c9e01917a8(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    for (int i = 0, j = n - 1; i <= j; ) {
        if (si + a[i] < sj + a[j]) {
            si += a[i++];
        } else {
            sj += a[j--];
        }
        min = Math.min(min, Tour.max(sum - si - sj, si, sj));
    }
    return min;
}

public static int func_fcf4c1cfea7540a985480c0441e84d27(int i, int
high, int low, long rest, long[] partials){
    int guess = (low + high) / 2;
    long second = partials[guess] - partials[i];
    if (second * 2 > rest)
        high = guess;
    else
        low = guess;
    return high;
}

public static double func_72794d71ad524f52896c85a00138bb59(int x1,
double remain, double x, double y, double[] len){
    remain -= (y + len[x1]) * (x1 - x) / 2;
    x = x1;
    return x;
}
```

```
public static double func_d16159251aad45ffabd8a52b11a08772(int i, int
n, int min, double ret){
    int max = n - 1;
    while (max - min > 3) {
        int left = (2 * min + max) / 3;
        int right = (2 * max + min) / 3;
        long leftCount = A.count(i, left);
        long rightCount = A.count(i, right);
        if (leftCount < rightCount) {
            max = right;
        } else {
            min = left;
        }
    }
    for (int j = min; j <= max; j++) {
        ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
    }
    return ret;
}

public static int func_11b0f79c8eaf4054b94510d6db6a1e7b(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
```

```
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return beforeHalf;
}

public static int func_002e3840b13d42ef819b82377e0838db(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    return x;
}

public static long func_521f476cc442478fa96e333b2bf12466(int lo, int
N, int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    return a;
}

public static long func_7f9b53f6bd494a9bb9b81c25243bfb71(int numtie,
int i, long waste, long initadd, long[] v){
    for (int j = i; j < 37; j++) {
        if (v[j] == v[i]) {
            numtie++;
        }
    }
    if (v[i] == v[i - 1]) {
        v[i]++;
        initadd++;
        waste += numtie;
```

```
    }
    long bet = i * (v[i] - 1);
    long min = 0;
    return min;
}

public static long[] func_1b968b34255448ad8160464d2d4c2036(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    return arr;
}

public static int[] func_cfdd0a3b3c3548968c9c8d48713cdafb(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    return data;
}

public static double func_f32bc1c3eb7641108ff816be53e7c1f6(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    for (int a = 0, b = 0; a < n; a++) {
        while (sum[b + 1] - sum[a] < sum[n] - sum[b + 1] && b < n) {
```

```
            b++;
        }
        for (int i = 0; i < 2; i++) {
            if (b - i < a)
                break;
            long s1 = sum[b - i + 1] - sum[a];
            long s2 = sum[a];
            long s3 = sum[n] - sum[b - i + 1];
            long smax = Math.max(s1, Math.max(s2, s3));
            res = Math.max(res, 1 - (double) smax / sum[n]);
        }
    }
    return res;
}

public static double func_20be2ebf382449e7a04d0a9b60af8954(int W, int
i, double cutArea, double[] cuts){
    double min = 0.0;
    double max = W + 0.00000001337;
    double mid = 0;
    while (max - min > 0.000000001) {
        mid = (min + max) / 2.0;
        double a = CodeJam_Round3_A.evalArea(mid);
        // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea, a);
        if (a > cutArea)
            max = mid;
        else
            min = mid;
    }
    cuts[i] = mid;
    return max;
}

public static long[] func_a184d8b74e2a4b77ad5c6a48ea717696(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    return rsum;
}

public static ArrayList func_9029ff8a9c794824a16ce0398732b1e6(double
X, double y0, double x0, ArrayList<Double> lx, ArrayList<Double> ly){
    lx.add(X);
    ly.add(-2000.0);
    x0 = lx.get(lx.size() - 1);
    y0 = ly.get(ly.size() - 1);
    return ly;
```

```
}

public static long[] func_92544afe5a494930a93a9a5a0667acc7(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    return psum;
}

public static long[] func_78c3735af471427b96bd99d7f3920d25(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    return a;
}

public static double func_064ea9e3c19f4d40ae279726927b5d8a(int W, int
U, int G, int t, Scanner input){
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    System.out.println("Case #" + t + ":");
    for (int i = 0; i < cuts.length; i++) System.out.println(cuts[i]);
    return g;
```

```
}

public static int func_c16f00a9f8074507a2c668ee3b54c7e6(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return y;
}

public static int func_1309aec3e3414b4091fb8a61349152d2(int n, int p,
int q, int r, Scanner sc){
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    return s;
}

public static double func_5782ce0de4794fdea645e88a00e1a218(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    return c;
}

public static long[] func_ea037e8ebb6144e9a8814de81ecc3d6c(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
```

```
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    return best;
}

public static int func_a1384633c80f4f55a829e57d8a32ce84(int py, int
px, int i, int[][] pos, double[] bound){
    int nx = pos[i][0];
    int ny = pos[i][1];
    double div = nx - px;
    for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
        bound[j] -= (ny - py) / div * (j - px) + py;
    }
    px = nx;
    py = ny;
    return nx;
}

public static long func_19626f379cb8479f90d366b4e9ed3353(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
```

```
    long subSum = 0;
    return subSum;
}

public static long func_7dcb9d74297a4589ac7af4ea1e326518(int index,
int i, int n, long x, long[] sum){
    while (index < n && sum[n] - sum[index + 1] > sum[index + 1] -
sum[i]) index++;
    x = Math.min(x, Math.max(sum[i], Math.max(sum[index] - sum[i],
sum[n] - sum[index])));
    return x;
}

public static long func_956e08dcb17a4cd0a14824bda3790c50(int low, int
i, int high, int n, long[] rsum){
    long prefixSum = (i == 0 ? 0 : rsum[i - 1]);
    while (high - low > 1) {
        int mid = (high + low) / 2;
        long sum = (rsum[mid] - prefixSum);
        long suffSum = rsum[n - 1] - sum - prefixSum;
        if (sum <= suffSum) {
            low = mid;
        } else {
            high = mid;
        }
    }
    return prefixSum;
}

public static double func_58992e9b699c49c980dc56cc5da85c84(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return ans;
}

public static long func_1b311de9c8c6430ca13b93735f58fc7d(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
```

```
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return min;
}

public static double func_391c3d67212c4ea8bea5ad26bb09887e(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    lastx = mid;
    return lastx;
}

public static long func_a334d2c7a63749189960cda876d72f3b(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
```

```
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return avg;
}

public static long func_013c9539b4ee4e0bbd030ea28dbb70d6(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    return lo;
}

public static Integer[] func_5317fdb2052c4876af86f30f515adc2f(int n,
int[] P, int[] L, Scanner in){
    Integer[] I = new Integer[n];
    for (int i = 0; i < n; i++) L[i] = in.nextInt();
    for (int i = 0; i < n; i++) {
        P[i] = in.nextInt();
        I[i] = i;
    }
    return I;
}

public static double func_55d1f0c062084ca1b5dcea1482734366(double
targetArea, double low, double curans, double high){
    while (low + 0.0000000001 < high) {
        double guess = (low + high) / 2;
        double a = A.area(curans, guess);
        if (a < targetArea) {
            low = guess;
        } else {
            high = guess;
```

```
        }
    }
    curans = (low + high) / 2;
    return curans;
}

public static double func_4278656204fe4e3f80e70adece95ded7(double t,
double k, double res, double d1, double dx){
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    return temp;
}

public static long[] func_c2e345f2cc02465b9e5092bd601b6ec3(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
```

```
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return arr;
}

public static long func_7c2911eaceec4483aa2c8bb7437f1506(int lo, int
mid, int hi, long s1, long left){
    long s2 = left - s1;
    if (s1 >= s2) {
        hi = mid;
    } else {
        lo = mid + 1;
    }
    return s2;
}

public static int func_7ad6f850afda41278612ead8b72f8a8f(int w, int u,
int l, Scanner in){
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
```

```
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    return x0;
}

public static long func_6ad774a820cc4d2e8213e184ca8280f3(int p, int q,
int n, int r, int s){
    TaskA.a = new long[n];
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    return minimal;
}

public static long[] func_a195657622ef4ed5811233df86a6e419(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
```

```
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return dp;
}

public static long[] func_8ed85e2004af4a86a9f88247143478a0(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    return psum;
}

public static long func_38553bba6c5943d58bb3f61d4b3740ef(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    for (int i = 0, j = n - 1; i <= j; ) {
        if (si + a[i] < sj + a[j]) {
            si += a[i++];
        } else {
            sj += a[j--];
        }
        min = Math.min(min, Tour.max(sum - si - sj, si, sj));
    }
    return sum;
}
```

```java
public static double func_5e3adae14b7745dea093522339e580d5(double k,
double d1, double want, double dx){
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return t;
}

public static int func_da58d90445f54300acf664f71beaa475(int caseNum,
PrintWriter pw, Scanner sc){
    pw.println("Case #" + caseNum + ":");
    int W = sc.nextInt();
    int L = sc.nextInt();
    int U = sc.nextInt();
    return U;
}

public static long[] func_56bffbedab944d1a82b778f5152a6b96(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    return a;
}

public static int func_4d870f3c953a4dd3816f3b4356cc3f1a(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return index;
}
```

```
public static long func_b560018a15ab4298aa43ab235fc11f0a(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return L;
}

public static int func_b250b1f020c5444ab556bf736751d3e7(int l, int g,
int u, int w, int[][] pos){
    double[] bound = new double[w + 1];
    int px = pos[0][0];
    int py = pos[0][1];
    for (int i = 1; i < l; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] -= (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    px = pos[l][0];
    py = pos[l][1];
    for (int i = l + 1; i < l + u; ++i) {
        int nx = pos[i][0];
        int ny = pos[i][1];
        double div = nx - px;
        for (int j = px + (i == l + 1 ? 0 : 1); j <= nx; ++j) {
            bound[j] += (ny - py) / div * (j - px) + py;
        }
        px = nx;
        py = ny;
    }
    double sum = 0.0;
    for (int i = 0; i < w; ++i) {
        sum += (bound[i] + bound[i + 1]) / 2.0;
    }
    double req = sum / g;
```

```
    double cur = 0.0;
    int p = 0;
    double used = 0.0;
    while (g > 1) {
        double a = bound[p];
        double b = bound[p + 1];
        double add = ((b - a) * used + a + b) * (1.0 - used) / 2.0;
        if (cur + add >= req) {
            double hi = 1.0;
            double lo = used;
            for (int j = 0; j < 100; ++j) {
                double mi = (hi + lo) / 2.0;
                double cand = ((b - a) * used + a + (b - a) * mi + a)
* (mi - used) / 2.0;
                if (cur + cand >= req) {
                    hi = mi;
                } else {
                    lo = mi;
                }
            }
            System.out.println(p + hi);
            cur = 0.0;
            used = hi;
            --g;
        } else {
            cur += add;
            ++p;
            used = 0.0;
        }
    }
    return p;
}

public static long[] func_49631dac35a34f13a0ad848c58657fcb(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    return a;
}

public static long func_013be7db4e1e458d8e0243a942139a22(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
```

```
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return sum;
}

public static long[] func_b405cd0209204db0a8ebecf22da25479(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    long[] pref = new long[n];
    for (int i = 0; i < n; i++) {
        pref[i] = ar[i];
        if (i > 0) {
            pref[i] += pref[i - 1];
        }
    }
    return ar;
}

public static long func_33e43da22b32496988efffe684bc1534(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
```

```
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return temp;
}

public static long func_516ae7e08b7849cd96e8febb5072c4fb(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    return temp;
}

public static int func_8b8412e2194f4a478fa93f3bc3c7017e(int taskIndex,
int N, long sum, long[] arr, long[] dp){
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
```

```
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    System.out.println(String.format("Case #%d: %.11f", taskIndex,
res));
    return second;
}

public static long[] func_b51e44286fc04b35b045c079d11c7bd6(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    return best;
}

public static long[] func_8f7eb2f326a547a2bcb4ec4d2f324b01(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    return a;
}

public static int func_0c6f2b0965a249a8b8fe375ce80023f0(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
```

```
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    return y0;
}

public static long[] func_6a05eaca55b54ac69071e528ad89f60f(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    return best;
}

public static long[] func_c060d5e3029b406dbcf68dde45ecf879(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    return a;
}

public static long func_5aeacd0677624f74b648890e2b0a1ebd(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
```

```
        } else {
            L = M;
        }
    }
    return tot;
}

public static long func_b8bd1fe7ef304eb3ae9d9b387424d09e(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return t1;
}

public static long func_011b43d864b04d9c88a1189a2b7a6f70(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    return sum;
}

public static long func_3abda7c796e241eb8f4e55f2b90a27b0(int lo, int
i, int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
```

```
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    return diff1;
}

public static double func_95614ec0f98440178abec7c830a314cb(int t, int
w, int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
```

```
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    return area;
}

public static long[] func_a014684cf13a4166a9debcfc056cc8e9(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    outer: while (low + 1 < high) {
        long mid = (low + high) / 2;
        int cur = 0;
        for (int i = 0; i < 3; i++) {
            long need = pref[cur] + mid;
            int x = Arrays.binarySearch(pref, need);
            if (x < 0) {
                x = -x - 2;
            }
            // System.out.println(x);
            if (x == n) {
                high = mid;
                continue outer;
            }
            cur = x;
        }
        low = mid;
    }
    // System.out.println(high);
    double ans = 1.0 * (pref[n] - high) / pref[n];
    return pref;
```

```
}

public static long[] func_a7ed55278d0944f78175629f3a3ac8da(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    return a;
}

public static long func_4be402073ca3485fadb3e9bbb8f2f736(int i, int n,
long l, long[] a, double max){
    long s = 0;
    long s2 = 0;
    for (int j = 0; j < n; j++) {
        if (j <= i) {
            s += (l - a[j]);
            s2 += (l - a[j]);
        } else if (a[j] < l + 1) {
            s += (l + 1 - a[j]);
        }
    }
    double w = s2 * (36.0 / (i + 1)) - s;
    // System.out.println(i + " " + l + " " + w);
    max = Math.max(max, w);
    return s2;
}

public static long[] func_cddb30e661c243a298066d6d66c5ca7e(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
```

```
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return a;
}

public static int[] func_9138ef52663e4e55ab738f841df9208a(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
```

```
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    return data;
}

public static double func_cc548a677a9e47e3bdab24e70dc39b4f(int
taskIndex, int N, long sum, long[] arr, long[] dp){
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    System.out.println(String.format("Case #%d: %.11f", taskIndex,
res));
    return res;
}

public static int func_44b3afd096a649b2b1c63e5c4f7db0bb(int n, Scanner
in){
    int p = in.nextInt(), q = in.nextInt(), r = in.nextInt(), s =
in.nextInt();
    long sum = 0;
    long[] a = new long[n];
```

```
    return s;
}

public static long func_2581fc10f1c3426fbd179d088b54d6e2(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    return sum;
}

public static TreeSet func_298176efec244c249b6fa425c65c108a(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return all;
}

public static double func_45796d32d3f449caa514fec7eddc5914(double mid,
double cutArea, double min, double max){
    double a = CodeJam_Round3_A.evalArea(mid);
    // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea, a);
    if (a > cutArea)
        max = mid;
    else
        min = mid;
    return max;
}
```

```
public static double func_bfd4c3f09d8441a992aa8ff5e9cfe2fe(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    lastx = mid;
    return lo;
}

public static long func_91651ff1df3c43d59c26772ebd5dbc8a(int n, long
P, long Q, long R, Scanner in){
    long S = in.nextInt();
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return ss;
}

public static a.Pair[] func_99cb0809fb9e40e39a7542ca47defd69(int n){
    a.Pair[] pairs = new a.Pair[n];
    int[] v1 = new int[n];
    return pairs;
}

public static double func_9476a190f34340058898d750cc6b30ad(double hi,
double cur, double used){
    cur = 0.0;
    used = hi;
    return used;
}

public static long[] func_b3826eb2333046849f5e4b8746fb562e(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
```

```
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return A;
}

public static long[] func_28e31f1cbc234ac69a45f91e0268a8de(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return LS;
}

public static int func_4fc37b357adc44a8b9ffe490c31da29f(int t, int w,
Scanner in){
    int l = in.nextInt();
    int u = in.nextInt();
    int g = in.nextInt();
```

```
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    return l;
}

public static double func_0cec6d8fb87648bf8cb50c6df7f0a541(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return dy0;
}

public static long func_533846a7e34445849f265bb29c56a908(long sum){
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return rest;
```

```
}

public static int
func_d890e6295def4af6ad0d8276339b9a7a(StringTokenizer st){
    int lastX = Integer.parseInt(st.nextToken());
    int bottom = Integer.parseInt(st.nextToken());
    return bottom;
}

public static long func_fe4e5f9a66ef4f2aa67fd65869e98c84(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    return t1;
}

public static long func_48551410ca5e454d88d8eaf0fc284bfb(int r, int s,
int n, int p, int q){
    long sum = 0;
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = (((long) i * p + q) % r + s);
        sum += a[i];
    }
    long min = sum;
    long si = 0, sj = 0;
    return min;
}

public static int func_8eadb3346d744662ad8f39d8ca200489(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
```

```
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return second;
}

public static long func_c52bf6a6344d4a2dbe75aebd25b06e87(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    return sum;
}

public static int func_b24c7cdb9a224e60ac8f0e197cd2c9ec(int t, int w,
int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
```

```
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
    return u;
}

public static long[] func_36518792d1e04fcd942e67c9fad39f07(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
        }
    }
    return partial;
```

```
}

public static int func_2289a426ef474a41acb066a31e349be1(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    return index;
}

public static long func_1cffa5f05101452299cd25ba34bfeb64(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
```

```
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return ss;
}

public static long func_61239b8cfd844f579b92871476afa70b(int i, int n,
long s, long l, long[] a){
    long s2 = 0;
    for (int j = 0; j < n; j++) {
        if (j <= i) {
            s += (l - a[j]);
            s2 += (l - a[j]);
        } else if (a[j] < l + 1) {
            s += (l + 1 - a[j]);
        }
    }
    double w = s2 * (36.0 / (i + 1)) - s;
    return s2;
}

public static int func_cd6c32d20ebb4b1687443dfb82ee8adc(int W, int U,
int L, int t, Scanner input){
    int G = input.nextInt();
    CodeJam_Round3_A.lower = new double[L][2];
    for (int i = 0; i < L; i++) {
        CodeJam_Round3_A.lower[i][0] = input.nextDouble();
        CodeJam_Round3_A.lower[i][1] = input.nextDouble();
    }
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
```

```
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    System.out.println("Case #" + t + ":");
    for (int i = 0; i < cuts.length; i++) System.out.println(cuts[i]);
    return G;
}

public static double func_fe8b92d896794ba7982219b21812767c(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    for (int i = 1; i <= n; i++) partials[i] = partials[i - 1] +
array[i - 1];
    long best = partials[n];
    for (int i = 0; i < n; i++) {
        long now = A.choose(partials, i, n);
        // System.out.println("  "+i+": "+now);
        best = Math.min(best, now);
    }
    long total = partials[n];
    return 1 - (double) (best) / total;
}

public static long func_65151ccfab444fb09c2d6f1bd123bf82(int n, int
first, long sum1, long mid, long[] a){
    while (first != n && sum1 < mid) {
```

```
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    return sum2;
}

public static long[] func_6286d7e532994deda6bdc0feb0dbc3fc(int n, int
p, Scanner sc){
    int q = sc.nextInt();
    int r = sc.nextInt();
    int s = sc.nextInt();
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    return a;
}

public static long[] func_f864f88e77dd4f6a83aea5f2067d6bfa(int n, int
p, int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    return a;
}

public static double func_c028848f8bd647c7a934e2958569d060(int ulidx,
int llidx, double rtn, double xe, double x1){
    // distance in x
    double d1 = x1 - A.u[ulidx].x;
    double d3 = xe - A.u[ulidx].x;
    double d2 = A.u[ulidx + 1].x - A.u[ulidx].x;
    double ysu = A.u[ulidx].y + (d1 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    double yeu = A.u[ulidx].y + (d3 / d2) * (A.u[ulidx + 1].y -
A.u[ulidx].y);
    d1 = x1 - A.l[llidx].x;
    d2 = A.l[llidx + 1].x - A.l[llidx].x;
    d3 = xe - A.l[llidx].x;
    double ysl = A.l[llidx].y + (d1 / d2) * (A.l[llidx + 1].y -
A.l[llidx].y);
    double yel = A.l[llidx].y + (d3 / d2) * (A.l[llidx + 1].y -
```

```
A.l[llidx].y);
    rtn += (xe - x1) * ((ysu - ysl) + (yeu - yel)) / 2;
    if (A.u[ulidx + 1].x < A.l[llidx + 1].x) {
        x1 = A.u[ulidx + 1].x;
        ulidx++;
    } else {
        x1 = A.l[llidx + 1].x;
        llidx++;
    }
    return yeu;
}

public static long func_92c5f4adf921410cad7e1d718538a348(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    return temp;
}

public static double func_a3d3a21e51c3478fb42568c9d10bdee8(int eq,
long[] init, long[] have){
    while (true) {
        if (eq + 1 < A.C && have[eq] == have[eq + 1]) {
            eq++;
        } else {
            break;
        }
    }
    double res = 0;
    for (int i = 0; i <= eq; i++) {
        res += (have[i] - init[i]) * 36;
    }
    return res;
```

```
}

public static int func_a9a4bb3e01f04608884f5057371ebef3(Scanner in){
    int N = in.nextInt();
    long p = in.nextInt();
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalf;
}

public static int func_b41a6abed55b46fb82005cc14b15a9ab(int j, int n,
int s2, int s1, int[] ps){
    int s3 = ps[n] - ps[j];
    int max = Math.max(s1, Math.max(s2, s3));
    return max;
}
```

```
public static long func_7ea5892f70514266abee90795f7a93b4(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    return sum;
}

public static long func_ef5e398d125c40e8a2005231665c1f15(int n, int N,
long k, long A, long[] is){
    long X = k * n - A;
    long Y = 0;
    for (int i = n; i < N; i++) Y += Math.max(0, k + 1 - is[i]);
    return Y;
}

public static int
func_f09853e926534f5cb6a8e198ce38dfc0(StringTokenizer st){
    int top = Integer.parseInt(st.nextToken());
    int cut = Integer.parseInt(st.nextToken());
    return top;
}

public static long[] func_52b460f12712409f9649be034b87464f(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
```

```
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return best;
}

public static long func_8d594a905e1f419a89e71a7f9dc122be(int r, int s,
int p, int q, int i){
    long curr = i - 1;
    curr *= p;
    curr += q;
    curr %= r;
    curr += s;
    A.num[i] = A.num[i - 1] + curr;
    return curr;
}

public static double func_151caf0549af4adcbafacecbc82b06a8(double k,
double d1, double want, double dx){
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return res;
}

public static int[] func_f99c1b5cfded4500b8b2af93471b53f3(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    double ans = 0;
    return ps;
}

public static long[] func_ae3bc1e892fc4cb68d4d9e65fbaa937f(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
```

```
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    return dp;
}

public static long[] func_90238f20b9934e8685fd1687ab34bf13(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return RS;
}

public static long func_018acc1d6ba6423b967ec1fc67e21b4e(int N, int
beforeHalf, long beforeHalfSum, long[] best, long[] A){
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
```

```
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalfSum;
}

public static double func_26fc546a1a154b5ab9cab27ac8ce4c39(int r, int
s, int n, int p, int q){
    A.num = new long[n + 1];
    for (int i = 1; i <= n; i++) {
        long curr = i - 1;
        curr *= p;
        curr += q;
        curr %= r;
        curr += s;
        A.num[i] = A.num[i - 1] + curr;
    }
    double ret = 0;
    for (int i = 0; i < n; i++) {
        int min = i;
        int max = n - 1;
        while (max - min > 3) {
            int left = (2 * min + max) / 3;
            int right = (2 * max + min) / 3;
            long leftCount = A.count(i, left);
            long rightCount = A.count(i, right);
            if (leftCount < rightCount) {
                max = right;
            } else {
                min = left;
            }
        }
        for (int j = min; j <= max; j++) {
            ret = Math.max(ret, 1 - A.count(i, j) * 1.0 / A.num[n]);
        }
    }
    return ret;
}

public static long func_a47dd765d0d14700a97d359c2994f55a(int n, int s,
int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
```

```
    return left;
}

public static long[] func_699a717ba60a4bca9852f8dff57d4d08(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    return best;
}

public static long[] func_e0340c1a5f444ca4a36b15d6630f3155(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
```

```
    int pow2 = 1;
    return LS;
}

public static long func_c5ef85d6b9e543f58c7c711b512066c4(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    return middle;
}

public static long[] func_60e246ec91e24121acfb2d5e5e9c865e(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    long best = 0;
    return nums;
}

public static long func_0829e85f615944f297ea2e8a56120bb2(int lo, int
i, int ans, int hi, long[] S){
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
```

```
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    return tot;
}

public static double func_3355f4f5c6054031a4952ac5c85b159c(double
target, double hi, double mid, double lastx, double lo){
    mid = (hi + lo) / 2;
    double ar = A.area(lastx, mid);
    if (ar > target)
        hi = mid;
    else
        lo = mid;
    return lo;
}

public static long func_b7e98d3008894b4aa28b666f5495cee0(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    while (pow2 * 2 <= N) {
        pow2 *= 2;
    }
    return sum;
}

public static double func_0a91effe9da64b5dbc38d4908d9a12c5(int G,
Integer[] xxx, int[][] y, int[][] x){
    int n = xxx.length;
    double[][] y1 = new double[2][n];
```

```
    int a = 0, b = 0;
    for (int i = 0; i < n; i++) {
        while (xxx[i] > x[0][a + 1]) a++;
        while (xxx[i] > x[1][b + 1]) b++;
        y1[0][i] = (y[0][a + 1] - y[0][a]) * 1.0 * (xxx[i] - x[0]
[a]) / (x[0][a + 1] - x[0][a]) + y[0][a];
        y1[1][i] = (y[1][b + 1] - y[1][b]) * 1.0 * (xxx[i] - x[1]
[b]) / (x[1][b + 1] - x[1][b]) + y[1][b];
    }
    double[] ans = new double[G - 1];
    double S = 0.0;
    return S;
}

public static long func_331309d309634c8785b5cc35c2329689(int i, int
ans, int n, long[] S){
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    return left;
}

public static long func_8d9c24a8468549a988ebbb90a168c115(int
winningThings, long payMoney, long low, long budget, long[] x){
    for (int i = 0; i < winningThings; i++) {
        payMoney += low - x[i];
    }
    if (payMoney > budget) {
        return -1;
    }
    --low;
    return payMoney;
}

public static long[] func_86b8f3b75fc44906bacda821654a71e7(int s, int
r, int p, int q, int n){
    int[] array = new int[n];
    for (int i = 0; i < n; i++) {
        array[i] = (i * p + q) % r + s;
    // System.out.print(" "+array[i]);
    }
    // System.out.println();
    long[] partials = new long[n + 1];
    return partials;
}

public static int func_fb53e19745074f888d958439d48a0a50(int i, int w,
```

```
double trazenaPovrsina, double[] res){
    double p = (i + 1) * trazenaPovrsina;
    double lg = 0;
    double dg = w;
    int counter = 0;
    while (lg + 0.00000001 < dg && counter < 1000) {
        counter++;
        double s = (lg + dg) / 2;
        double tp = C.povrsinaDo(s);
        if (tp < p)
            lg = s;
        else
            dg = s;
    }
    if (counter == 1000)
        System.out.println((dg - lg) + " " + lg + " " + dg);
    res[i] = (lg + dg) / 2;
    return counter;
}

public static int func_2b43a1ff71d0429d87bc5ebbf02d6843(int N, long q,
long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    return j;
}

public static int func_46826e1840c5469e9819d21f4bff7650(int min, int
h, int max, long[] imos){
    int med1 = (min * 2 + max) / 3;
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    return med1;
}

public static int func_e584fdbbfe8240c389b5cda2c411d2cf(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    return beforeHalf;
```

```
}

public static long func_9523e12defa9491ab8d56bc2b782c9b6(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    return tmp;
}

public static long func_1be9071a661e46dbaaefa45534ccf17c(int n, long
p, long q, long s, long r){
    for (int y = 0; y < n; y++) {
        A.array[y + 1] = A.array[y] + (y * p + q) % r + s;
    }
    int index = 0;
    long min = Long.MAX_VALUE;
    return min;
}

public static long[] func_418f366e6226483587c818eaadf209b9(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
```

```
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return arr;
}

public static double func_c5b4e9a64ec34da0b09881c95bfbef1b(double
target, double hi, double lastx, double lo){
    double mid = 0;
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    return hi;
}

public static int func_7fd8cb0610564b54a0546dfbe773cf03(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    int Tba = 100 * Lb + La * (100 - Fb);
    return Tab;
}

public static double func_6dadc6551fb24c6cbb1ae3ab78267cc9(int l,
double bx, double ans, double ax, double ay){
    double by = A.highy[l + 1];
    if (ax > A.highx[l]) {
        ay = A.highy[l] + (ax - A.highx[l]) / (A.highx[l + 1] -
A.highx[l]) * (A.highy[l + 1] - A.highy[l]);
    }
    if (bx < A.highx[l + 1]) {
        by = A.highy[l + 1] + (bx - A.highx[l + 1]) / (A.highx[l] -
A.highx[l + 1]) * (A.highy[l] - A.highy[l + 1]);
    }
    ans -= (bx - ax) * (1000 - (ay + by) / 2);
    return by;
}

public static long func_97a416a0e9a9405d86ee876471697123(int n, long
P, long S, long Q, long R){
```

```
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
    return l;
}

public static long[] func_a838242b70ba433ca50fd2c0e950c192(int n, long
s, long p, long q, long r){
    long[] nums = new long[n];
    for (int i = 0; i < n; i++) {
        nums[i] = (i * p + q) % r + s;
    }
```

```
    long[] rsum = new long[n];
    for (int i = 0; i < n; i++) {
        if (i != 0) {
            rsum[i] += (rsum[i - 1]);
        }
        rsum[i] += nums[i];
    }
    return rsum;
}

public static int func_424918e3fb0c4e8aa1d31862e57f2c02(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    return beforeHalf;
}

public static double func_55336e916cc04abcadaf34c600718ccd(double t,
double k, double res, double d1, double dx){
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
```

```
    return res;
}

public static long func_d3c5d8b249ca4714b9dfb33d913e3cf6(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return t2;
}

public static long[] func_ef7990f4b34148dcb310f019536e58d6(int N, long
r, long s, long p, long q){
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    final long total = sums[N];
    return sums;
}

public static long func_544fb35382484c199fb18a1303f5d8a4(long mid,
long min, long budget, long max, long[] bets){
    long need = 0;
    for (int i = 0; i < bets.length; i++) {
        if (bets[i] < mid) {
            need += mid - bets[i];
        }
    }
    if (need > budget) {
        max = mid;
    } else {
        min = mid;
    }
    return min;
}

public static long func_3c9f6d3815aa4ffeadc8ac601f209bb4(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
```

```
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return sum;
}

public static int[] func_f6e843bc17214cc3be1845f2411a558f(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    return arr;
}

public static int func_0cd3f8064a8f4662b2b101db03935b2a(int x0, double
remain, double x, double[] len, PrintWriter out){
    x0 = (int) x;
    int x1 = x0 + 1;
    double k = 1.0 * (len[x1] - len[x0]);
    double y = k * (x - x0) + len[x0];
    while (remain > (y + len[x1]) * (x1 - x) / 2) {
        remain -= (y + len[x1]) * (x1 - x) / 2;
        x = x1;
        x0 = (int) x;
        x1 = x0 + 1;
        k = 1.0 * (len[x1] - len[x0]);
        y = k * (x - x0) + len[x0];
    }
```

```
    if (Math.abs(k) > 10e-6) {
        double a = k;
        double b = 2 * (y - k * x);
        double c = k * x * x - 2 * x * y - 2 * remain;
        double d = Math.sqrt(b * b - 4 * a * c);
        double x2 = (d - b) / (2 * a);
        double x3 = -1.0 * (d + b) / (2 * a);
        if (x2 > x && x2 <= x1) {
            out.println(A.format(x2));
            x = x2;
        } else {
            out.println(A.format(x3));
            x = x3;
        }
    } else {
        x += remain / y;
        out.println(A.format(x));
    }
    return x0;
}

public static long func_8a570660f1a743a0a30bd2b3a62bf7f1(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    return L;
}

public static long func_42ad16be72104360afa781feaf3d7f8b(int lo, int
i, int n, long max, long[] S){
    int hi = i;
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
```

```
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return left;
}

public static long func_fc5cc2c2220041e5b097c327adfa6e10(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    return temp;
}

public static int func_9053118114f1421ca30bbc3ff459025a(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
```

```
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    return y;
}

public static long[] func_f09d2f88cab14e55bb9ae6d3ec27bb8f(int N, int
b, int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    return RS;
}

public static double func_fe1bf345e2f942198dbb531128cf84ef(int u, int
l, int tt, int g, Scanner in){
    int[] x1 = new int[l], y1 = new int[l], x2 = new int[u], y2 = new
int[u];
    for (int i = 0; i < l; i++) {
        x1[i] = in.nextInt();
        y1[i] = in.nextInt();
    }
    for (int i = 0; i < u; i++) {
        x2[i] = in.nextInt();
        y2[i] = in.nextInt();
    }
    double area = 0;
    for (int i = 0; i < l - 1; i++) {
        area += x1[i] * y1[i + 1] - x1[i + 1] * y1[i];
    }
    area += x1[l - 1] * y2[u - 1] - x2[u - 1] * y1[l - 1];
    for (int i = u - 1; i >= 1; i--) {
        area += x2[i] * y2[i - 1] - x2[i - 1] * y2[i];
    }
    area += x2[0] * y1[0] - x1[0] * y2[0];
    area = Math.abs(area) / g;
    int p1 = 0, p2 = 0;
    double cx = 0;
    System.out.println("Case #" + tt + ":");
    for (int i = 0; i < g - 1; i++) {
        double ca = area;
        while (ca > 1e-9) {
            // System.out.println (p1 + " " + p2 + " " + ca);
            double nx = Math.min(x1[p1 + 1], x2[p2 + 1]);
```

```
            double a1 = y1[p1 + 1] - y1[p1];
            double b1 = x1[p1] - x1[p1 + 1];
            double c1 = a1 * x1[p1] + b1 * y1[p1];
            double a2 = y2[p2 + 1] - y2[p2];
            double b2 = x2[p2] - x2[p2 + 1];
            double c2 = a2 * x2[p2] + b2 * y2[p2];
            for (int j = 0; j < 100; j++) {
                double cy1, cy2, my1, my2, mx = (nx + cx) * 0.5;
                cy1 = (c1 - a1 * cx) / b1;
                my1 = (c1 - a1 * mx) / b1;
                cy2 = (c2 - a2 * cx) / b2;
                my2 = (c2 - a2 * mx) / b2;
                double la = (cy2 - cy1 + my2 - my1) * (mx - cx);
                // System.out.println (" --> " + la + " " + cx + " " +
mx  + " " + cy1 + " " + cy2 + " " + my1 + " " + my2);
                if (la < ca) {
                    cx = mx;
                    ca -= la;
                } else {
                    nx = mx;
                }
            }
            if (ca > 1e-9) {
                if (x1[p1 + 1] < x2[p2 + 1])
                    ++p1;
                else if (x1[p1 + 1] > x2[p2 + 1])
                    ++p2;
                else {
                    ++p1;
                    ++p2;
                }
            }
        }
        System.out.println(cx);
    }
    return cx;
}

public static long func_a5c97aef946241b19a58f460049c17b4(int N, long
sum, long avg, long[] arr, long[] dp){
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
```

```
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return temp;
}

public static int func_836be92635af4f8eb853ad13e59cd8c5(int N, long p,
Scanner in){
    long q = in.nextInt();
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
```

```
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    return beforeHalf;
}

public static long[] func_e47e9f6f48264301bdebddd127eceef7(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    while (L + 1 < R) {
        long M = (L + R) >> 1;
        // out.println(L+" "+R+" "+M+" "+ok(M,cum));
        if (A.ok(M, cum)) {
            R = M;
        } else {
            L = M;
        }
    }
    // out.println(R+" "+tot);
    double ans = (tot - R) * 1.0 / tot;
    return cum;
}

public static int func_a22cf581577b42d2b666299e28a2ca60(int t, int w,
int l, Scanner in){
    int u = in.nextInt();
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
```

```
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    return u;
}

public static long func_039863e593314f568b5210f3fc675335(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return leftSum;
}

public static long[] func_4a1efb12497f48f9b653bad35ee0f946(int p, int
q, int n, Scanner in){
    int r = in.nextInt();
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
```

```
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return S;
}

public static long func_f48a9eb9dd72498bb8cbf2e1f9cf0bde(int lo, int
i, int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    return center;
}

public static long[] func_ee6f668d451f46a686cfbf963a5df881(int N, long
r, long p, long s, long q){
    long[] amt = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        amt[i] = (((i - 1) * p + q) % r) + s;
    }
    long[] psum = new long[N + 1];
    for (int i = 1; i <= N; i++) {
        psum[i] = psum[i - 1] + amt[i];
    }
    return psum;
}

public static int func_23c6666ecf554af2bab454d38c62d621(int N, long p,
```

```
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    return first;
}

public static long func_8b71fa8dead84e06b565d9934a0ed295(int n, int r,
int q, int s, int p){
    int[] values = new int[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        values[i] = (int) (((long) i * p + q) % r + s);
        sum += values[i];
    }
    long res = Integer.MIN_VALUE, leftSum = 0, middle = values[0],
rightSum = sum - middle;
    for (int a = 0, b = 0; a < n; a++) {
        while (true) {
            if (b + 1 == n)
                break;
            if (b < a || middle + values[b + 1] <= rightSum - values[b
+ 1]) {
                middle += values[b + 1];
                rightSum -= values[b + 1];
                b++;
            } else
                break;
        }
        res = Math.max(res, sum - Math.max(leftSum, Math.max(rightSum,
middle)));
        if (b + 1 < n) {
            res = Math.max(res, sum - Math.max(leftSum,
Math.max(rightSum - values[b + 1], middle + values[b + 1])));
        }
        leftSum += values[a];
        middle -= values[a];
    }
    return rightSum;
}

public static int func_fa898aa522644b4b841cf2e9c79e833c(int n, int p,
```

```
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    return x;
}

public static long func_12eb90407af1488188856936bc43f224(int N, int b,
int d, int c, long a){
    int[] arr = new int[N];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = (int) ((i * a + b) % c + d);
        sum += arr[i];
    }
    long[] LS = new long[N];
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    int pow2 = 1;
    while (pow2 * 2 <= N) {
        pow2 *= 2;
    }
    return LO;
}

public static double func_5446b8b493584ab0af6ea4b6e362a26a(double k,
double d1, double want, double dx){
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
```

```
    }
    return res;
}

public static long[] func_8cb216d1733241f28985fcf390bbd82d(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    return best;
}

public static int func_0f67297494c946dd9d441365d8294a5b(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    return beforeHalf;
}

public static TreeSet func_c6762599bf0646cf9272b733a02024b6(int r, int
p, int s, int q, int n){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < a.length; i++) {
        a[i] = (1L * i * p + q) % r + s;
        sum += a[i];
    }
    TreeSet<Long> all = new TreeSet<Long>();
    all.add(0L);
    long x = 0;
    return all;
}

public static long[] func_bdd3ccd66adb42a8bcb4553add334a96(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    return partial;
}

public static int func_c459c1e65b9c48ea86fccf5d5b7a7397(int n, int p,
```

```
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    return x;
}

public static long func_04b9391b3532479794245b074424849b(int n, long
t2, long sum, long t1, long[] a){
    long rest = sum;
    int j = 0;
    double ans = 0;
    for (int i = 0; i < n; i++) {
        while (j < n && t2 + a[j] <= rest - a[j]) {
            t2 += a[j];
            rest -= a[j];
            j++;
        }
        double pp = 1.0 * (sum - Math.max(t1, Math.max(t2, rest))) /
sum;
        ans = Math.max(ans, pp);
        if (j < n) {
            pp = 1.0 * (sum - Math.max(t1, Math.max(t2 + a[j], rest -
a[j]))) / sum;
            ans = Math.max(ans, pp);
        }
        t1 += a[i];
        t2 -= a[i];
        if (j == i) {
            rest -= a[j];
            j++;
        }
    }
    return rest;
}
```

```
public static long func_6655a4df1913495ba5aeac0153024767(int n, long
right, long left, long[] a, long[] sum){
    long mid = (left + right) / 2;
    int last = 0;
    boolean ok = false;
    long sum1 = 0;
    int first = 0;
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
        second--;
    }
    if (sum1 >= mid && sum2 >= mid) {
        if (second >= first) {
            ok = true;
        } else {
            long total = 0;
            if (second != -1)
                total += sum[second];
            if (first != n)
                total += sum[n - 1] - sum[first - 1];
            if (total >= mid)
                ok = true;
        }
    }
    return sum2;
}

public static long func_ec0010c1edcd4232a7a5a4a3b0e9cc81(int i, int
ans, int n, long max, long[] S){
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    long take = S[n] - Math.max(center, Math.max(left, right));
    max = Math.max(max, take);
    return center;
}
```

```java
public static int[] func_570d476af4004a1e8b1a4831435a635c(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    return qty;
}

public static double func_0fc8b98911dc451d94f6c1dcc9aa8524(double
chunkLeft, double thisLow, double currentLow, double thisHigh, double
currentHigh){
    currentHigh = thisHigh;
    currentLow = thisLow;
    chunkLeft = 0;
    return currentLow;
}

public static PrintWriter func_731c85439b694d6087d1303c1dcb20e6(int
caseNum, PrintWriter pw, Scanner sc){
    System.out.println("Processing case " + caseNum);
    pw.println("Case #" + caseNum + ":");
    int W = sc.nextInt();
    int L = sc.nextInt();
    int U = sc.nextInt();
    return pw;
}

public static double func_b7209f69dece4346ade4a5efb47cfe37(int n,
double[] y2, double[] yy, double[] y1){
    for (int i = 0; i < n; i++) yy[i] = y2[i] - y1[i];
    double ss = 0;
    return ss;
}

public static long func_70419a32ab414c61a1cf02d4667898db(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
```

```
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return ans;
}

public static int[] func_cb56350707f8447184eae33bf4f7bb24(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    return a;
}

public static long func_31ddd31220a34264af756a19d54db2d0(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return R;
}

public static long func_7dabbc6ce9a54dbfaa0d2d8feddd0658(int p, int q,
int n, int r, int s){
    for (int i = 0; i < n; i++) {
        TaskA.a[i] = ((long) i * p + q) % r + s;
    }
    for (int i = 1; i < n; i++) TaskA.a[i] += TaskA.a[i - 1];
    long minimal = TaskA.a[n - 1];
    for (int i = 0; i < n; i++) {
```

```
        minimal = Math.min(minimal, Math.max(TaskA.sum(0, i),
TaskA.sum(i, n)));
    }
    for (int i = 2, j = 1; i < n; i++) {
        while (j + 1 < i && TaskA.sum(0, j + 1) <= TaskA.sum(j + 1,
i)) {
            ++j;
        }
        minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i, n),
TaskA.sum(j, i)), TaskA.sum(0, j)));
        if (j + 1 < i) {
            minimal = Math.min(minimal, Math.max(Math.max(TaskA.sum(i,
n), TaskA.sum(j + 1, i)), TaskA.sum(0, j + 1)));
        }
    }
    return minimal;
}

public static long func_d67bae36fd2f472094a175377e30d229(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return sum;
}

public static long func_a51d54221ae5474d8a645dffb2e7665d(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
```

```
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return min;
}

public static long func_00b301836afc4b4f88358d60f9c84728(int n, int r,
int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    return t2;
}

public static int func_7db80a41e0b4462b96de2b4c0ee0ff89(int n, int[]
xl, int[] yl, int[] x, double[] y1){
    int ind = 0;
    for (int i = 0; i < n; i++) {
        while (x[i] > xl[ind + 1]) ind++;
        y1[i] = yl[ind] + (double) (yl[ind + 1] - yl[ind]) * (x[i] -
xl[ind]) / (xl[ind + 1] - xl[ind]);
    }
    double[] y2 = new double[n];
    return ind;
}

public static long func_6ef96b69b53e492a89359ae90affdf82(int first,
int N, long avg, long temp, long[] arr){
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
```

```
        ++index;
    }
    return temp;
}

public static long func_e9772a4fcca94ccaacbe8ba8986b9136(int N, long
q, long p, Scanner in){
    long r = in.nextInt();
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return beforeHalfSum;
}

public static double func_d7ff5aca5b3f49debe2b9465a9fb03dd(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
```

```
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    lastx = mid;
    return mid;
}

public static int func_14f08a6c30cb43feb46e162e07385b16(int N, long q,
long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    return j;
}

public static long[] func_014ff7dfc5924662b9d2d6927da05ef7(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return arr;
}

public static int[] func_bafda404b2864366a901c1914850997e(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
```

```
    return qty;
}

public static long func_c0870a5b8a1d4ece8b7934bf3e4d6db7(int n, int r,
int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    // System.out.println(Arrays.toString(pref));
    // System.out.println(Arrays.toString(a));
    // can't
    long low = 0;
    // can
    long high = pref[n];
    return low;
}

public static long[] func_dc3d4c40e4a74d04871ecf18aa401ff7(int n, int
s, int q, int p, int r){
    long[] a = new long[n];
    for (int i = 0; i < n; i++) {
        a[i] = ((i * 1L * p + q) % r + s);
    }
    long[] sum = new long[n];
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    while (right - left > 1) {
        long mid = (left + right) / 2;
        int last = 0;
        boolean ok = false;
        long sum1 = 0;
        int first = 0;
        while (first != n && sum1 < mid) {
            sum1 += a[first];
            first++;
        }
        int second = n - 1;
        long sum2 = 0;
        while (second != -1 && sum2 < mid) {
            sum2 += a[second];
            second--;
        }
        if (sum1 >= mid && sum2 >= mid) {
            if (second >= first) {
```

```
                ok = true;
            } else {
                long total = 0;
                if (second != -1)
                    total += sum[second];
                if (first != n)
                    total += sum[n - 1] - sum[first - 1];
                if (total >= mid)
                    ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
    }
    return a;
}

public static int func_40e5cab76957432785b30bb1bfb51071(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return first;
}

public static double[] func_cb218c7d0d6242b881769e760eb12acb(int w,
int G, double tS, double[] h, double[] S){
    for (int i = 0; i < w; i++) {
        S[i] = (h[i] + h[i + 1]) / 2;
        tS += S[i];
    }
    double[] ans = new double[G - 1];
```

```
    double curS = 0, eps = 1e-9, meps = 1e-19;
    return ans;
}

public static int func_3747c0cc76984b82afea16d0d09c8204(int w, int l,
double[] low, Scanner in){
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    return y0;
}

public static double[] func_2299a68cd03048c5bd91f912ace4a0e2(int w,
int u, int g, int l, Scanner in){
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    for (int i = 1; i < u; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
```

```
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            upper[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    for (int i = 0; i <= w; i++) {
        len[i] = upper[i] - low[i];
    }
    double area = 0;
    for (int i = 1; i <= w; i++) {
        area += (len[i - 1] + len[i]) / 2;
    }
    area /= g;
    return low;
}

public static long[] func_520f223e1c664b75bc8bd84d5ef75fb8(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    long L = 0;
    long R = tot + 1;
    return arr;
}

public static double func_7c64384677d14e619bbbe7eff7faa814(int w, int
G, double[] h, double[] S){
    double tS = 0;
    for (int i = 0; i < w; i++) {
        S[i] = (h[i] + h[i + 1]) / 2;
        tS += S[i];
    }
    double[] ans = new double[G - 1];
    double curS = 0, eps = 1e-9, meps = 1e-19;
    return tS;
}

public static long[] func_10040f9376144794ad6f6c75a5a2e998(int N, long
r, long s, long p, long q){
```

```
    final long[] counts = new long[N];
    for (int i = 0; i < N; i++) counts[i] = (i * p + q) % r + s;
    final long[] sums = new long[N + 1];
    for (int i = 0; i < N; i++) sums[i + 1] = sums[i] + counts[i];
    return sums;
}

public static int[] func_68529dd1c517443e9375b982c9ae688a(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return data;
}

public static int[] func_cb143d7014574152b86c22b0b31e28b9(int n, int
r, int s, int q, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (int) (((long) i * p + q) % r + s);
    }
    long[] pref = new long[n + 1];
    for (int i = 0; i < n; i++) {
        pref[i + 1] = pref[i] + a[i];
    }
    return a;
}

public static long func_0def182d5ae04e49bba4a7620ebb92c3(int lo, int
N, int i, int hi, long[] psum){
    long left = psum[N] - psum[i];
    while (lo < hi) {
        int mid = (lo + hi) >> 1;
        long s1 = psum[mid] - psum[i];
        long s2 = left - s1;
```

```
        if (s1 >= s2) {
            hi = mid;
        } else {
            lo = mid + 1;
        }
    }
    long t = psum[lo] - psum[i];
    long a = Math.max(t, left - t);
    if (lo > i + 1) {
        t = psum[lo - 1] - psum[i];
        a = Math.min(a, Math.max(t, left - t));
    }
    return t;
}

public static int func_d3ececdbc06046569cd26b612cd5ede1(int a, int b){
    // System.out.println("TEST");
    // System.out.println(area);
    ArrayList<Double> res = new ArrayList<Double>();
    // System.out.println(slice);
    a = 0;
    b = 0;
    return b;
}

public static long[] func_83b4231bd8ed400db96e3bb21048d8a6(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    long answer = partial[count];
    for (int i = 0; i < count; i++) {
        long first = partial[i] - partial[0];
        if (first > answer)
            continue;
        long remaining = partial[count] - first;
        int at = Arrays.binarySearch(suffixes, remaining >> 1);
        if (at >= 0)
            answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at], remaining - suffixes[at])));
        else {
            int at1 = -at - 1;
            int at2 = -at - 2;
            if (at1 >= 0 && at1 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at1], remaining - suffixes[at1])));
```

```
            if (at2 >= 0 && at2 <= count)
                answer = Math.min(answer, Math.max(first,
Math.max(suffixes[at2], remaining - suffixes[at2])));
        }
    }
    return partial;
}

public static int func_5c3bfafb434343c68f2977337bb8a180(int t, int w,
int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    return x0;
}

public static int func_842ddb85698e4702a4223ecda4c64a03(int x0,
A2.Pair p2, A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return x1;
}

public static double func_f96cfca632cd49de9572ea4b772ac585(A2.Pair p2,
A2.Pair p1){
    int x0 = p1.a;
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
```

```
    double crt = (dy0 + dy1) * (x1 - x0) / 2;
    return crt;
}

public static double func_2a5a9e2640b94d43a0da3231d938894e(double
chunkLeft, double thisLow, double currentLow, double thisHigh, double
currentHigh){
    currentHigh = thisHigh;
    currentLow = thisLow;
    chunkLeft = 0;
    return currentHigh;
}

public static int func_5d2924e615ac46c585e0f2f73e5ac0b1(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    return index;
}

public static int func_45c7cec6cdea4df09882524986fe3c36(int u, int n,
int l, int[] xl, int[] x){
    for (int i = 1; i < l - 1; i++) x[i - 1 + u] = xl[i];
    Arrays.sort(x);
    double[] y1 = new double[n];
    int ind = 0;
    return ind;
}

public static double func_cde06f7f3ebe4da8bbe991fcf18f24e4(double
target, double hi, double mid, double lastx, double lo){
    while (hi - lo > 0.00000000001) {
        mid = (hi + lo) / 2;
        double ar = A.area(lastx, mid);
```

```
        if (ar > target)
            hi = mid;
        else
            lo = mid;
    }
    System.out.println(mid);
    return mid;
}

public static long[] func_f072da5eb2544cbf817b094da83b79ff(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    long[] pref = new long[n];
    for (int i = 0; i < n; i++) {
        pref[i] = ar[i];
        if (i > 0) {
            pref[i] += pref[i - 1];
        }
    }
    long max = Long.MAX_VALUE;
    for (int rght = 0; rght < n; rght++) {
        long rs = pref[n - 1] - pref[rght];
        long sum = pref[rght];
        int left = -1, right = rght;
        while (left < right - 1) {
            int mid = (left + right) >> 1;
            if (pref[mid] * 2 >= sum) {
                right = mid;
            } else {
                left = mid;
            }
        }
        for (int t = right - 2; t <= right + 2; t++) {
            if (0 <= t && t <= rght) {
                long ans = rs;
                ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
                max = Math.min(max, ans);
            }
        }
    }
    return ar;
}

public static int[] func_ed95902582774323b5eba530a20c3c9a(int n, int
p, int q, int r, int s){
    int[] t = new int[n];
    long[] sum = new long[n + 1];
```

```
    for (int i = 0; i < n; i++) {
        t[i] = (int) ((i * (long) p + q) % r + s);
        sum[i + 1] = sum[i] + t[i];
    }
    double res = 0;
    for (int a = 0, b = 0; a < n; a++) {
        while (sum[b + 1] - sum[a] < sum[n] - sum[b + 1] && b < n) {
            b++;
        }
        for (int i = 0; i < 2; i++) {
            if (b - i < a)
                break;
            long s1 = sum[b - i + 1] - sum[a];
            long s2 = sum[a];
            long s3 = sum[n] - sum[b - i + 1];
            long smax = Math.max(s1, Math.max(s2, s3));
            res = Math.max(res, 1 - (double) smax / sum[n]);
        }
    }
    return t;
}

public static long[] func_51bf164bd2c74c5d9b687fd6abca8933(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return arr;
}

public static double func_d1f1c361764142f79e3a5d48340e57a2(double hi,
double lo, double b, double a, double used){
    double mi = (hi + lo) / 2.0;
```

```
    double cand = ((b - a) * used + a + (b - a) * mi + a) * (mi -
used) / 2.0;
    return mi;
}

public static double func_d7156b594e3f4100ae629196d67a62f9(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    return res;
}

public static double[] func_baa96857b024455193216871c6889654(int t,
int w, int u, int l, Scanner in){
    int g = in.nextInt();
    if (t == 5) {
        System.out.println(w + " " + l + " " + u + " " + g);
    }
    double[] len = new double[w + 1];
    double[] low = new double[w + 1];
    int x0 = in.nextInt();
    int y0 = in.nextInt();
    low[x0] = y0;
    for (int i = 1; i < l; i++) {
        int x = in.nextInt();
        int y = in.nextInt();
        double k = 1.0 * (y - y0) / (x - x0);
        for (int j = x0 + 1; j <= x; j++) {
            low[j] = k * (j - x0) + y0;
        }
        x0 = x;
        y0 = y;
    }
    double[] upper = new double[w + 1];
    x0 = in.nextInt();
    y0 = in.nextInt();
    upper[x0] = y0;
    return low;
}

public static int func_fe139b075f64404ab17d1541b62980b1(int N, long q,
long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
```

```
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return beforeHalf;
}

public static long[] func_9b78de1d30f044308236fda376871897(int n, int
r, int s, int q, long p){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * p + q) % r) + s;
        sum += a[i];
    }
    long t1 = 0;
    long t2 = 0;
    long rest = sum;
    int j = 0;
    double ans = 0;
    return a;
}

public static double func_976d20eb9ce643449fd7415165d2b9e7(int l,
double ans, double right, double left){
```

```
    double ax = Math.max(A.lowx[l], left);
    double bx = Math.min(A.lowx[l + 1], right);
    double ay = A.lowy[l];
    double by = A.lowy[l + 1];
    if (ax > A.lowx[l]) {
        ay = A.lowy[l] + (ax - A.lowx[l]) / (A.lowx[l + 1] -
A.lowx[l]) * (A.lowy[l + 1] - A.lowy[l]);
    }
    if (bx < A.lowx[l + 1]) {
        by = A.lowy[l + 1] + (bx - A.lowx[l + 1]) / (A.lowx[l] -
A.lowx[l + 1]) * (A.lowy[l] - A.lowy[l + 1]);
    }
    ans -= (bx - ax) * ((ay + by) / 2 + 1000);
    return ax;
}

public static double[] func_1364dc86fd224079b3c7e2f9ff43d89a(int u,
int n, int l, int[] xl, int[] x){
    for (int i = 1; i < l - 1; i++) x[i - 1 + u] = xl[i];
    Arrays.sort(x);
    double[] y1 = new double[n];
    return y1;
}

public static int func_f1f1ce895fba4c65b41f500464d63676(int N, long q,
long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
```

```
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    return beforeHalf;
}

public static long func_46edbcf57e544b089f43ef9032c1e470(int t, long
ans, long sum, long max, long[] pref){
    ans = Math.max(ans, Math.max(sum - pref[t], pref[t]));
    max = Math.min(max, ans);
    return max;
}

public static long func_f06d401cc706487f92d3d473467fd181(int a, int b,
long[] cum){
    if (b < a)
        return 0;
    if (a == 0)
        return cum[b];
    else {
        return cum[b] - cum[a - 1];
    }
}

public static long[] func_88fff342edf34b009dd0f90398d4cf87(int r, int
p, int q, int n, Scanner in){
    int s = in.nextInt();
    int[] A = new int[n];
    long[] S = new long[n + 1];
    for (int i = 0; i < n; i++) {
        A[i] = (int) (((long) i * p + q) % r + s);
        S[i + 1] = S[i] + A[i];
    }
    long max = 0;
    return S;
}

public static long func_7151bfdb852f4fa08aa485a7851cdc56(int min, int
h, int max, int med1, long[] imos){
    int med2 = (min + max * 2) / 3;
    long v1 = ProblemA.take(imos, h, med1);
    long v2 = ProblemA.take(imos, h, med2);
    if (v1 < v2) {
        min = med1;
    } else {
        max = med2;
    }
    return v2;
```

```
}

public static long func_112a2cd473cb4a01a3d662b14097e393(int p2, int
p3, int l2, int l3, long l1){
    long eCur = (l1 + l2) * p2 + (l1 + l2 + l3) * (100 - p2) * p3;
    long eSwap = (l1 + l3) * p3 + (l1 + l2 + l3) * (100 - p3) * p2;
    return eCur;
}

public static long func_fe42813ac50941baaa31387ee9402276(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    return sum;
}

public static long func_87074d1a07134ca5ad611f27739dc2c7(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    return ss;
}

public static int func_6ecf60ab98a64e21b5f33a4398f6a76b(int n, long
mid, long[] a){
    int last = 0;
    boolean ok = false;
    long sum1 = 0;
    int first = 0;
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
```

```
        second--;
    }
    return last;
}

public static double func_0415e409557849fda255d43ae3f2ea1b(int W, int
U, int G, int t, Scanner input){
    CodeJam_Round3_A.upper = new double[U][2];
    for (int i = 0; i < U; i++) {
        CodeJam_Round3_A.upper[i][0] = input.nextDouble();
        CodeJam_Round3_A.upper[i][1] = input.nextDouble();
    }
    double g = G;
    double[] cuts = new double[G - 1];
    double firstArea = CodeJam_Round3_A.evalArea(W - 0.00000001);
    // System.out.println(firstArea);
    for (int i = 0; i < cuts.length; i++) {
        double cutArea = (i + 1) / g * firstArea;
        double min = 0.0;
        double max = W + 0.00000001337;
        double mid = 0;
        while (max - min > 0.000000001) {
            mid = (min + max) / 2.0;
            double a = CodeJam_Round3_A.evalArea(mid);
            // System.out.printf("%.1f %.2f %.5f\n" , mid, cutArea,
a);
            if (a > cutArea)
                max = mid;
            else
                min = mid;
        }
        cuts[i] = mid;
    }
    System.out.println("Case #" + t + ":");
    for (int i = 0; i < cuts.length; i++) System.out.println(cuts[i]);
    return g;
}

public static int[] func_82d89c001e6a45e292daec40802f700c(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    return a;
}

public static double func_1b1ef9b1e46442e0a216c0ecb3f4a0ab(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
```

```
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return answer;
}

public static int func_ccf3be7597d04bfba8bf255f1be3a4cf(int G, int[][]
y, int[][] x, TreeSet<Integer> xx){
    Integer[] xxx = xx.toArray(new Integer[] {});
    int n = xxx.length;
    double[][] y1 = new double[2][n];
    int a = 0, b = 0;
    for (int i = 0; i < n; i++) {
        while (xxx[i] > x[0][a + 1]) a++;
        while (xxx[i] > x[1][b + 1]) b++;
        y1[0][i] = (y[0][a + 1] - y[0][a]) * 1.0 * (xxx[i] - x[0]
[a]) / (x[0][a + 1] - x[0][a]) + y[0][a];
        y1[1][i] = (y[1][b + 1] - y[1][b]) * 1.0 * (xxx[i] - x[1]
[b]) / (x[1][b + 1] - x[1][b]) + y[1][b];
    }
    double[] ans = new double[G - 1];
```

```
    double S = 0.0;
    return b;
}

public static long func_4e4dccfc7b434b40b4b78e4db38ce540(int N, long
r, long p, long s, long q){
    long[] arr = new long[N];
    long[] cum = new long[N];
    long tot = 0;
    for (int i = 0; i < N; i++) {
        arr[i] = ((i * p + q) % r) + s;
        if (i > 0)
            cum[i] = cum[i - 1] + arr[i];
        else
            cum[i] = arr[i];
        tot += arr[i];
    }
    return tot;
}

public static long func_060f89a929264bcb81b0bf66c75d726a(int n, long[]
sum, long[] a){
    for (int i = 0; i < n; i++) sum[i] = a[i] + (i == 0 ? 0 : sum[i -
1]);
    long left = 0, right = (long) 1e15;
    return left;
}

public static int[] func_75f23e609c734922b4cc74ffd1aa2942(int s, int
n, int q, int r, int p){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = (i * p + q) % r + s;
    }
    int[] ps = new int[n + 1];
    for (int i = 1; i <= n; i++) {
        ps[i] = ps[i - 1] + a[i - 1];
    }
    double ans = 0;
    return a;
}

public static long[] func_084606e9068147daaf32dafa09dcf14d(int N, long
q, long p, long r, Scanner in){
    long s = in.nextInt();
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
```

```
    int beforeHalf = 0;
    long beforeHalfSum = 0;
    long sum = 0;
    for (int n = 0; n < N; n++) {
        sum += A[n];
        while (beforeHalf < n && beforeHalfSum + A[beforeHalf] < sum /
2) {
            beforeHalfSum += A[beforeHalf];
            beforeHalf++;
        }
        long max = sum;
        if (beforeHalf > 0) {
            max = Math.max(beforeHalfSum, sum - beforeHalfSum);
        }
        if (beforeHalf <= n - 1) {
            long value = Math.max(beforeHalfSum + A[beforeHalf], sum -
beforeHalfSum - A[beforeHalf]);
            max = Math.min(max, value);
        }
        best[n] = max;
    }
    long min = best[N - 1];
    long subSum = 0;
    for (int n = N - 1; n >= 1; n--) {
        subSum += A[n];
        min = Math.min(min, Math.max(subSum, best[n - 1]));
    }
    double answer = (sum - min) / (double) sum;
    return best;
}

public static int func_2fb0c5cef0a844b0b115e1fd9c0cf026(int La, int
Fa, GameLevel another){
    int Lb = another.timeCost;
    int Fb = another.failProb;
    int Tab = 100 * La + Lb * (100 - Fa);
    return Tab;
}

public static long func_26bc0681e6d549c690247e79cdd91fb9(int
winningThings, long payMoney, long low, long budget, long[] x){
    if (payMoney > budget) {
        return -1;
    }
    --low;
    long high = budget + 1;
    while (high - low > 1) {
        long mid = low + high >> 1;
        if (A.canDo(budget, x, winningThings, mid)) {
            low = mid;
```

```
        } else {
            high = mid;
        }
    }
    return low;
}

public static double func_55ed6cb80cdd4e50bd47138a00b2d5ad(double d2,
double d1, double want, double dx){
    double k = (d2 - d1) / dx;
    if (Math.abs(k) < 1e-9) {
        return want / d1;
    }
    double t = Math.sqrt(Math.max(d1 * d1 + 2 * k * want, 0));
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
    }
    return res;
}

public static double func_e60220eb23da4ceda1e8dfa6dc3b8290(int n, int
p, int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    double res = 1. * lo / sum;
    System.out.printf("%.9f\n", 1 - res);
```

```
    return res;
}

public static long func_67a503e86c84459d999c5e0508c938a0(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return ss;
}

public static int[] func_f2ed71c8cdd34adf8ef3d83f8dde0016(int n){
    int[] len = new int[n];
    int[] p = new int[n];
    return len;
}

public static long func_8504ffa1463241a5a042635507ec80c3(int
winningThings, long high, long low, long budget, long[] x){
    long mid = low + high >> 1;
    if (A.canDo(budget, x, winningThings, mid)) {
        low = mid;
    } else {
        high = mid;
    }
    return high;
}

public static double func_0f05ae60a51e4a5e9138ec3426284d91(double t,
double k, double d1, double want, double dx){
    double res = (t - d1) / k;
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    // double res = k < 0 ? (-t - d1) / k : (t - d1) / k;
    double temp = (d1 + (d1 + k * res)) * .5 * res;
    if (Math.abs(temp - want) > 1e-5) {
        System.out.println("WTF ");
        System.out.println(temp + " = " + want + " " + res);
    }
    return res;
}

public static long func_8c56de228d1143cda3c32d703a6d6b7f(int i, int
```

```
ans, int n, long diff1, long[] S){
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    return right;
}

public static long func_b230b3dc8e834411885db8a2b4d19c77(int numtie,
int i, long bet, long waste, long tp){
    waste += numtie * tp;
    bet += i * tp;
    return bet;
}

public static int func_eca7e17c3f0f4161af5e65b6de9ef356(A2.Pair p2,
A2.Pair p1){
    int x1 = p2.a;
    double dy0 = p1.yh - p1.yl;
    double dy1 = p2.yh - p2.yl;
    return x1;
}

public static long[] func_b4d7ab4173b54b2ab7bcf5d947c57268(int N, long
q, long p, long r, long s){
    long[] A = new long[N];
    for (int n = 0; n < N; n++) {
        A[n] = (n * p + q) % r + s;
    }
    long[] best = new long[N];
    return best;
}

public static long[] func_1bb9fa2f64cb4f05a55556362e8f3253(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    long ans = A[N];
    for (int i = 0; i < N; i++) {
        long third = A[N] - A[i + 1];
        while (j < i && A[i + 1] - A[j + 2] > A[j + 2]) j++;
        {
```

```
            long first = A[j + 1];
            long second = A[i + 1] - A[j + 1];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
        if (j < i) {
            long first = A[j + 2];
            long second = A[i + 1] - A[j + 2];
            ans = Math.min(ans, Math.max(first, Math.max(second,
third)));
        }
    }
    return A;
}

public static int[] func_a92c203b43424ef29732cc6fb039f4c3(int q, int
s, int p, int r, int n){
    int[] a = new int[n];
    for (int i = 0; i < n; i++) a[i] = (int) ((i * (long) p + q) % r +
s);
    // sout(a);
    long[] sum = new long[n + 1];
    sum[0] = 0;
    for (int i = 0; i < n; i++) sum[i + 1] = sum[i] + a[i];
    long x = sum[n];
    int index = 0;
    return a;
}

public static long[] func_d1eab302e5e84c03a0248ad199633a8d(int n, long
p, long q, long r, long s){
    long[] ar = new long[n];
    for (int i = 0; i < n; i++) {
        ar[i] = (i * p + q) % r + s;
    }
    long[] pref = new long[n];
    return ar;
}

public static long func_b60df4438be6463ab91bb57ae02e3577(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    return r;
```

```
}

public static long[] func_0730db5c507c4736b6d7c35813ca63e2(int r, int
count, int q, int s, long p){
    int[] qty = new int[count];
    for (int i = 0; i < count; i++) qty[i] = (int) ((i * p + q) % r +
s);
    long[] partial = ArrayUtils.partialSums(qty);
    long[] suffixes = new long[count + 1];
    for (int i = 0; i <= count; i++) suffixes[i] = partial[count] -
partial[count - i];
    return suffixes;
}

public static int func_e23ef73e43d0439584903a174e69e821(int low, int
high, int n, long prefixSum, long[] rsum){
    int mid = (high + low) / 2;
    long sum = (rsum[mid] - prefixSum);
    long suffSum = rsum[n - 1] - sum - prefixSum;
    if (sum <= suffSum) {
        low = mid;
    } else {
        high = mid;
    }
    return low;
}

public static int func_133091b2b4e14c04bbcca6c32a1d054e(int lo, int i,
int n, int hi, long[] S){
    int ans = -1;
    long tot = S[i + 1] - S[0];
    while (lo <= hi) {
        int mid = (lo + hi) / 2;
        long center = S[i + 1] - S[mid];
        long left = tot - center;
        // System.out.println(mid + " " + left + " " + center);
        if (left > center) {
            hi = mid - 1;
        } else {
            ans = mid;
            lo = mid + 1;
        }
    }
    // System.out.println(i + " " + ans);
    long diff1 = Math.abs(S[i + 1] - S[ans] - (S[ans] - S[0]));
    long diff2 = Math.abs(S[i + 1] - S[ans + 1] - (S[ans + 1] -
S[0]));
    if (diff2 < diff1)
        ans++;
    // System.out.println(i + " " + ans + " " + diff1 + " " +
```

```
    // diff2);
    long center = S[i + 1] - S[ans];
    long left = S[ans] - S[0];
    long right = S[n] - left - center;
    return lo;
}

public static long[] func_b9541b47627e404b912070ca7b04fde7(int n, long
P, long S, long Q, long R){
    long[] a = new long[n];
    long ss = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((i * P + Q) % R + S);
        ss += a[i];
    }
    long l = 0;
    long r = ss;
    while (r > l + 1) {
        long m = (l + r) / 2;
        boolean ok = true;
        for (int i = 0; i < n; i++) {
            if (a[i] > m)
                ok = false;
        }
        if (!ok) {
            l = m;
        } else {
            int c = 0;
            long q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            q = 0;
            while (c < n && q + a[c] <= m) {
                q += a[c];
                c++;
            }
            if (c == n) {
                r = m;
            } else {
                l = m;
            }
        }
    }
```

```java
    return a;
}

public static double func_672444f0181b4aa89d1b8be6e3e9a936(double t,
double k, double res, double d1, double dx){
    while (res < -1e-6 || res > dx + 1e-6) {
        res = (-t - d1) / k;
        System.out.println("ASDSAD");
    }
    return res;
}

public static long func_9b30807d261f4d66ab9a372baf8a36ef(int N, int[]
arr, long sum, long[] LS){
    long[] RS = new long[N];
    for (int i = 0; i < N; ++i) {
        LS[i] = arr[i] + (i > 0 ? LS[i - 1] : 0);
        RS[i] = arr[N - i - 1] + (i > 0 ? RS[i - 1] : 0);
    }
    long LO = 0;
    long HI = sum;
    return HI;
}

public static double func_29a7fa889f6a451ab148fc52b6d5579a(double k,
double remain, double x, double y, double a){
    double b = 2 * (y - k * x);
    double c = k * x * x - 2 * x * y - 2 * remain;
    double d = Math.sqrt(b * b - 4 * a * c);
    double x2 = (d - b) / (2 * a);
    return c;
}

public static long[] func_003ce5c73bd047b6b29e2b60184ca0ef(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
```

```
        first = index;
        ++index;
    }
    temp = 0;
    return arr;
}

public static long[] func_daac0f076a0647c6bd8664ef9be5613b(int N, long
q, long s, long r, long p){
    long[] a = new long[N];
    for (int i = 0; i < N; i++) a[i] = ((i * p + q) % r + s);
    long[] A = new long[N + 1];
    for (int i = 0; i < N; i++) A[i + 1] = A[i] + a[i];
    int j = 0;
    return a;
}

public static long func_f40c8059aa354cec92c175f5cf9628b3(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    return temp;
}

public static int func_2170db7c639b43759453aca5646464ca(int n, int p,
int q, int s, int r){
    long[] a = new long[n];
    long sum = 0;
    for (int i = 0; i < n; i++) {
        a[i] = ((long) i * p + q) % r + s;
        sum += a[i];
    }
```

```
    int x = 0, y = 0;
    long tmp = 0;
    for (int i = 0; i < n; i++) {
        tmp += a[i];
        if (tmp * 3 <= sum) {
            x = i;
        }
        if (tmp * 3 <= sum * 2) {
            y = i;
        }
    }
    double ans = 0;
    for (int dx = -3; dx <= 3; dx++) {
        for (int dy = -3; dy <= 3; dy++) {
            int rx = x + dx;
            int ry = y + dy;
            if (rx < 0 || rx >= n || ry < 0 || ry >= n) {
                continue;
            }
            long sum1 = 0, sum2 = 0, sum3 = 0;
            for (int i = 0; i < n; i++) {
                if (i < rx) {
                    sum1 += a[i];
                } else {
                    if (i <= ry) {
                        sum2 += a[i];
                    } else {
                        sum3 += a[i];
                    }
                }
            }
            long maxSum = Math.max(sum1, Math.max(sum2, sum3));
            double rate = (sum - maxSum) * 1.0 / sum;
            if (rate > ans) {
                ans = rate;
            }
        // out.write(maxSum + " " + sum + "\n");
        }
    }
    return x;
}

public static int func_765f7c9d3bd84ef8a28884b3b322e61b(int N, long p,
long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
```

```
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    return second;
}

public static boolean func_0e67a7191ef143d28e1ada7317720e21(int n,
long right, long left, long[] a, long[] sum){
    long mid = (left + right) / 2;
    int last = 0;
    boolean ok = false;
    long sum1 = 0;
    int first = 0;
    while (first != n && sum1 < mid) {
        sum1 += a[first];
        first++;
    }
    int second = n - 1;
    long sum2 = 0;
    while (second != -1 && sum2 < mid) {
        sum2 += a[second];
        second--;
    }
    if (sum1 >= mid && sum2 >= mid) {
        if (second >= first) {
            ok = true;
        } else {
            long total = 0;
            if (second != -1)
                total += sum[second];
            if (first != n)
                total += sum[n - 1] - sum[first - 1];
            if (total >= mid)
```

```
                ok = true;
            }
        }
        if (ok) {
            left = mid;
        } else {
            right = mid;
        }
        return ok;
    }

    public static int func_da25840bff55422fbac35ebb33606b73(long[] a,
    Scanner in){
        int nn = in.nextInt();
        for (int i = 0; i < nn; i++) {
            a[i] = in.nextLong();
        }
        Arrays.sort(a);
        double max = 0;
        return nn;
    }

    public static int func_6ab6817d22aa462081b87410241ec3d9(int N, long p,
    long r, long s, long q){
        long[] arr = new long[N];
        long[] dp = new long[N + 1];
        long sum = 0;
        for (int i = 0; i < N; ++i) {
            arr[i] = ((i * p + q) % r) + s;
            sum += arr[i];
            dp[i + 1] = dp[i] + arr[i];
        }
        long avg = sum / 3;
        int first = 0;
        int second = N - 1;
        long temp = 0;
        int index = 0;
        while (temp < avg && index < N) {
            temp += arr[index];
            first = index;
            ++index;
        }
        temp = 0;
        index = first + 1;
        while (temp < avg && index < N) {
            temp += arr[index];
            second = index;
            ++index;
        }
        double res = 0;
```

```
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return first;
}

public static long func_213df4d6d6e94866acfd6d8021f94fb3(int N, long
p, long r, long s, long q){
    long[] arr = new long[N];
    long[] dp = new long[N + 1];
    long sum = 0;
    for (int i = 0; i < N; ++i) {
        arr[i] = ((i * p + q) % r) + s;
        sum += arr[i];
        dp[i + 1] = dp[i] + arr[i];
    }
    long avg = sum / 3;
    int first = 0;
    int second = N - 1;
    long temp = 0;
    int index = 0;
    while (temp < avg && index < N) {
        temp += arr[index];
        first = index;
        ++index;
    }
    temp = 0;
    index = first + 1;
    while (temp < avg && index < N) {
        temp += arr[index];
        second = index;
        ++index;
    }
    double res = 0;
    for (int i = Math.max(0, first - 100); i < Math.min(N, first +
100); ++i) {
        for (int j = Math.max(0, second - 100); j < Math.min(N, second
+ 100); ++j) {
            if (j <= i) {
```

```
                continue;
            }
            long a = dp[i + 1];
            long b = dp[j] - dp[i + 1];
            long c = dp[N] - dp[j];
            long left = sum - Math.max(a, Math.max(b, c));
            res = Math.max(res, 1.0 * left / sum);
        }
    }
    return avg;
}

public static long[] func_bed55fb6b46d42cb9e2f1f4e974e3c3c(int n, int
r, int s, int p, int q){
    long[] sum = new long[n + 1];
    for (int i = 0; i < n; i++) {
        sum[i + 1] = sum[i] + ((long) i * p + q) % r + s;
    }
    return sum;
}

public static long func_ceeef8a9b28746e7ab851b7377f64bcc(int n, int p,
int q, int r, int s){
    int[] data = new int[n];
    for (int i = 0; i < n; i++) {
        data[i] = (int) (((long) i * p + q) % r + s);
    }
    long lo = 1, hi = (long) 1e13;
    while (lo < hi - 1) {
        long mid = (lo + hi) / 2;
        // System.out.println(mid);
        if (a.canDo(data, mid))
            hi = mid;
        else
            lo = mid;
    }
    if (!a.canDo(data, lo))
        lo++;
    // System.out.println(lo);
    long sum = 0;
    for (int i = 0; i < n; i++) sum += data[i];
    return lo;
}

public static int func_8f6030832b5d4d7ead82357e7182956d(int j, int i,
int n, int s1, int[] ps){
    int s2 = ps[j] - ps[i];
    int s3 = ps[n] - ps[j];
    int max = Math.max(s1, Math.max(s2, s3));
    return s3;
```

```
}

public static long func_7b6b32b1332b404aacc2a46f3db9df85(Scanner in){
    int N = in.nextInt();
    long p = in.nextInt();
    return p;
}

public static double[] func_d80adaaccfac4ea6b199c8b311cbeb2c(int n,
int[] x, double[] y2, double[] y1){
    /*                   for (int i = 0; i < n; i++) out.print(x[i] +
" ");
    out.println();
    for (int i = 0; i < n; i++) out.print(y1[i] + " ");
    out.println();
    for (int i = 0; i < n; i++) out.print(y2[i] + " ");
    out.println();*/
    double[] yy = new double[n];
    for (int i = 0; i < n; i++) yy[i] = y2[i] - y1[i];
    double ss = 0;
    for (int i = 0; i < n - 1; i++) {
        ss += (x[i + 1] - x[i]) * (yy[i] + yy[i + 1]) / 2;
    }
    return yy;
}

public static double func_c98b2586e98148c6bfcf5091dd1412f7(int w){
    double l = 0;
    double r = w;
    return r;
}
```