ルータを見つける (routers)

左右に伸びた l メートルの長さの道路があり, n 個の小さなルータが, この道路上のそれぞれ異なる場所に設置されている. 道路の左端を **原点** と呼ぶ. ルータには左から順に 0 から $n-1$ までの番号が付けられており, ルータ i は原点から $p[i]$ メートルの場所に設置されている.

ルータ 0 は原点に設置されていることが保証されており, それ以外のルータについても, 原点からちょうど**偶数**メートルの地点に設置されている.

あなたは n 個のルータのそれぞれの位置を知りたい. ルータは非常に小さく遠くから視認することができないので, ルータの位置を知るために以下の手法を用いることにした:

- 検出器を原点から x メートル離れた地点に設置し,
- その検出器に最も近いルータの番号を得る. 検出器に最も近いルータが 2 つある場合には, 小さい方の番号を得る.

あなたは検出器を q 回まで用いることができる. すべてのルータの位置を知るための戦略を考案せよ.

実装の詳細

あなたは以下のプロシージャを実装しなさい:

```
int[] find_routers(int l, int n, int q)
```

- l: 道路の長さ.
- n: ルータの数.
- q: 検出器を用いることができる回数.
- このプロシージャは, 採点プログラムによりちょうど 1 回だけ実行される.
- このプロシージャは, 配列 p を戻り値として返す. これは各ルータの位置を示すもので, $p[i]$ は, ルータ i と原点との距離である必要がある.

上記のプロシージャから, 以下のプロシージャを呼び出すことができる:

```
int use_detector(int x)
```

- x: 検出器を設置する位置と原点との距離.
- x は 0 以上 l 以下でなければならない.
- このプロシージャは, 設置した検出器に最も近いルータの番号を戻り値として返す. 検出器に最も近いルータが 2 つある場合には, 小さい方の番号を返す.

- このプロシージャを q 回を超えて呼び出すことはできない.

入出力例

入出力例 1

以下の呼び出しを考えてみる:

```
find_routers(5, 2, 10)
```

2 つのルータが長さ 5 メートルの道路上にあり, あなたは 10 回まで `use_detector` を呼び出すことができる. 2 つのルータは, 原点からそれぞれ 0 メートルと 4 メートルの位置にあるとしよう.

`find_routers` から, `use_detector(3)` と呼び出すと, 1 を戻り値として得る. これは, 原点から 4 メートルの位置にあるルータ 1 が検出器に最も近いからである.

`find_routers` から, `use_detector(2)` と呼び出すと, 0 を戻り値として得る. これは, ルータ 0 とルータ 1 が検出器に最も近い 2 つのルータであり, これらのうちルータ 0 の方が番号が小さいからである.

以上で十分な情報が出揃っており, 2 つのルータはそれぞれ原点から 0 メートルと 4 メートルの位置にあると分かる.

従って, `find_routers` は戻り値として $[0, 4]$ を返すべきである.

入出力例 2

以下の呼び出しを考えてみる:

```
find_routers(6, 3, 10)
```

3 つのルータが長さ 6 メートルの道路上にあり, あなたは 10 回まで `use_detector` を呼び出すことができる. 3 つのルータは, 原点からそれぞれ 0 メートル, 2 メートルと 6 メートルの位置にあるとしよう.

`find_routers` から, `use_detector(5)` と呼び出すと, 2 を戻り値として得る. これは, 原点から 6 メートルの位置にあるルータ 2 が検出器に最も近いからである.

`find_routers` から, `use_detector(4)` と呼び出すと, 1 を戻り値として得る. これは, ルータ 1 とルータ 2 が検出器に最も近い 2 つのルータだからである.

以上で十分な情報が出揃っており, 3 つのルータはそれぞれ原点から 0 メートル, 2 メートルと 6 メートルの位置にあると分かる.

従って, `find_routers` は戻り値として $[0, 2, 6]$ を返すべきである.

制約

- $p[0] = 0$
- $0 \leq p[i] \leq l$ であり, $p[i]$ は偶数. $(0 \leq i \leq n-1)$
- $p[i] < p[i+1]$ $(0 \leq i \leq n-2)$

小課題

1. (16 点) $l = 100\ 000$, $n = 2$, $q = 100\ 001$
2. (21 点) $l = 100\ 000$, $n = 100$, $q = 100\ 001$
3. (23 点) $l = 100\ 000$, $n = 2$, $q = 20$
4. (40 点) $l = 100\ 000$, $n = 1000$, $q = 20\ 000$

加えて, 小課題 4 では次の規則に従って部分点が付く.

`use_detector` の呼び出し回数の全テストケース中の最大値を m とする.

- $m > 20\ 000$ の場合, あなたの得点は 0 点である.
- $7500 < m \leq 20\ 000$ の場合, あなたの得点は $\frac{20\ 000 - m}{12\ 500} \cdot 40$ 点である.
- $m \leq 7500$ の場合, あなたの得点は 40 点である.

採点プログラムのサンプル

採点プログラムのサンプルは, 以下の形式で入力を読み込む:

- 1 行目: l n q
- 2 行目: $p[0]$ $p[1]$ $\ldots$ $p[n-1]$

採点プログラムのサンプルは, 以下の形式であなたの答えを出力する.

- 1 行目: `find_routers` の戻り値 $p[0]$ $p[1]$ $\ldots$ $p[n-1]$.
- 2 行目: `use_detector` を呼び出した回数.