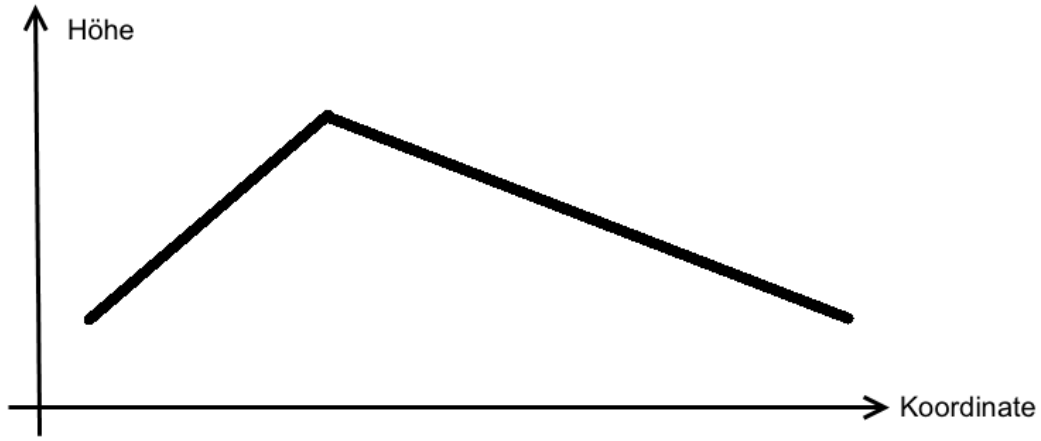
Wandertour

Hans und Gerald sind zwei begeisterte Wanderer. Eines Tages haben sie die Idee den Großglockner zu erklimmen. Die beiden verlieren keine Zeit und starten am nächsten Morgen los auf den Berg. Leider hat keiner in der Eile daran gedacht, sich die Wetterprognosen vorher anzusehen und somit werden sie vom Schneesturm und Nebel komplett überrascht. Nachdem sie ein paar Stunden im wilden Schneetreiben umherspazierten, wusste keiner mehr von Ihnen, wo sie sich gerade befinden. Da haben die beiden plötzlich eine Idee:

Hans, der begeisterte Technik-Fan, hat seine neue Höhenmessuhr mit, mit der man die Meereshöhe der aktuellen Position feststellen kann. Gerald hat heuer ein Projekt an der Universität betreut, welches Informationen über die Meereshöhe einer bestimmten Koordinate liefert. Man sendet einfach die Koordinate per SMS und bekommt die Meereshöhe zurück.

Das Problem scheint gelöst! Jedoch haben die beiden nicht mehr viel Guthaben auf ihrem Telefon um beliebig viele SMS senden zu können. Schreibe ein Programm, welches mit einer begrenzten Anzahl an Abfragen ihre Position feststellt.

Das Höhenprofil des Großglockners kann man sich wie folgt vorstellen: Es gibt nur eine Koordinate entlang dieser sich die beiden befinden können. Auf beiden Seiten des Gipfels steigt die Höhe streng monoton an, d.h. zwei benachbarte Koordinaten besitzen eine unterschiedliche Höhe. Die folgende Grafik soll dies verdeutlichen:



Achtung:

Bei diesem Beispiel sind Tokens im am Server aktiviert. Wenn ihr Code einsendet, seht ihr euer Ergebnis nur, wenn ihr dafür einen Token einsetzt. Jeder Teilnehmer hat *genau* 3 Tokens und bekommt im Verlauf der Qualifikation *keine* dazu. Ihr müsst euer Programm also *selber testen*. Es ist trotzdem möglich beliebig viele Einsendungen zu machen. Euer Score ist das Maximum aus allen Submission mit Token (den Score, den ihr seht) und der *letzten* Submission ohne Token.

Implementierungsdetails

Schreibe ein Programm welches die folgende Routine implementiert:

```
int findCoordinate(int N, int height)
```

Diese Routine wird am Anfang des Programms nur einmal aufgerufen. Parameter:

- `N` ist die Anzahl an Koordinaten. Die Koordinaten gehen von 1 bis inklusive `N`.
- `height` ist die Höhe, auf der sich die beiden befinden.

Rückgabewert:

- Die Koordinate bei der sich die beiden befinden. Falls es mehrere Koordinaten gibt, die dieselbe Höhe wie Hans und Gerald haben, reicht es eine beliebige Koordinate mit dieser Höhe zurückzugeben.

Für die Höhenabfrage kannst du folgende Funktion aufrufen:

```
int getHeight(int coordinate)
```

Diese Routine liefert gibt die Meereshöhe für die gegebene Koordinate zurück. Folgende Bedingung muss eingehalten werden, andernfalls wird das Programm beendet: $1 \leq$ `coordinate` $\leq$ `N`

Du darfst die Methode `getHeight` nur *150 Mal* aufrufen. Für mehr Abfragen reicht das Guthaben auf dem Telefon nicht aus. Wenn du diese Anzahl der Abfragen überschreitest, dann wird das Programm beendet und das Ergebnis als falsch gewertet.

Du musst genau eine Datei einreichen, `HIKING.cpp`. Diese Datei muss das oben beschriebene Unterprogramm implementieren und die oben beschriebene Signatur respektieren. Dieses Unterprogramm muss sich wie oben beschrieben verhalten. Es ist dir natürlich freigestellt, andere Unterprogramme zur internen Anwendung anzulegen.

Dein eingeschicktes Programm darf in keiner Weise mit der Standard Ein- und Ausgabe oder mit irgendeiner Datei interagieren. Ein und Ausgabe werden von einem Grader übernommen. Du kannst am Server alle notwendigen Dateien als Attachment herunterladen.

Eingabe

Der Grader liest die Eingabe in folgendem Format ein. Zeile 1: `N` und `height` durch ein Leerzeichen getrennt. In jeder der folgenden `N` Zeilen befindet sich eine Ganzzahl, die die Messhöhe repräsentiert.

Ausgabe

Der Grader gibt je nach Ergebnis verschiedene Texte aus:

- Falls die Anzahl der erlaubten Aufrufe von getHeight überschritten worden sind wird der Text “Too many queries” ausgegeben.

- Falls der Wert, der der Funktion `getHeight` übergeben wird ungültig ist, wird eine entsprechende Fehlermeldung ausgegeben.
- Falls der Wert, der von der Funktion `findCoordinate` zurückgeliefert wird ungültig ist, wird eine entsprechende Fehlermeldung ausgegeben.
- Falls das Ergebnis richtig ist, wird "Correct" ausgegeben.
- Falls das Ergebnis falsch ist, wird "Incorrect" ausgegeben.

Beispiele

Eingabe	Beispielinteraktion	Anmerkungen
5 10 2 4 6 8 10	getHeight(2) = 4 getHeight(4) = 8 getHeight(5) = 10	Beispiel für Subtask 2. In diesem Fall soll findCoordinate 5 zurückgeben.

Eingabe	Beispielinteraktion	Anmerkungen
5 10 2 4 10 8 4	getHeight(2) = 4 getHeight(4) = 8 getHeight(3) = 10	Beispiel für Subtask 3. In diesem Fall soll findCoordinate 3 zurückgeben.

Eingabe	Beispielinteraktion	Anmerkungen
6 4 2 4 10 8 4 3	getHeight(1) = 2 getHeight(4) = 8 getHeight(5) = 4	Beispiel für Subtask 4. In diesem Fall soll findCoordinate 2 oder 5 zurückgeben.

Subtasks

Allgemein gilt:

- $1 \leq \texttt{N} \leq 10^6$
- $1 \leq$ alle Höhenangaben $\leq 10^7$

Subtask 1 (0 Punkte): Beispieleingaben

Subtask 2 (24 Punkte): Der Gipfel befindet sich auf der Koordinate `N`.

Subtask 3 (18 Punkte): `N` ist ungerade. Der Gipfel befindet sich genau in der Mitte.

Subtask 4 (58 Punkte): Keine Einschränkungen

Limits

Zeitlimit: 1 s **Speicherlimit:** 256 MB