

Feladat: TRI

Háromszögek

CEOI 2018, nap 2. Memória limit: 256 MB.

16.08.2018

Byteland egy szép ország, ahol n ($n \geq 3$) város van. A városok a sík n darab különböző pontjával adottak, amelyeket 1-től n -ig sorszámozunk. Nem ismerjük a városok koordinátáit, de tudjuk hogy semelyik három pont nem esik egy egyenesre.

Egy n pontból álló ponthalmaz konvex burka az a legkisebb területű olyan konvex poligon, amely a ponthalmaz minden pontját vagy a belsejében tartalmazza, vagy az oldalára esik. Egy konvex poligon minden szöge kisebb, mint 180 fok, és oldalai nem metszik egymást.

Kiszámítandó a városokat tartalmazó konvex burok csúcsainak száma. A feladat megoldásához kérdéseket tehetsz fel. Egy kérdés három **különböző** város sorszámát tartalmazza: i, j, k ($1 \leq i, j, k \leq n$). A válasz a három pont által meghatározott háromszög körüljárási irányát adja meg. Igaz értéket ad, ha az i, j, k pontok sorrendje órajárással megegyező, egyébként hamisat.

Kommunikáció

A programodnak egy könyvtári modult kell használnia, amely a kérdéseket és a válaszadást valósítja meg.

A modul (`trilib.h` C és C++ nyelvekre) a következő függvényeket tartalmazza:

- `int get_n();`
A városok számát adja.
- `bool is_clockwise(int a, int b, int c);`
`true` értéket ad, ha az a, b, c ($1 \leq a, b, c \leq n, a \neq b \neq c \neq a$) pontokkal meghatározott háromszög körüljárása órajárással megegyező, és `false` értéket ad, ha órajárással ellentétes.
- `void give_answer(int s);`
Ezzel kell közölnöd a kiszámítandó eredményt, azaz s a konvex burok csúcsainak száma.

A `trilib` Java osztály az allábbi metódusokat tartalmazza:

- `static public int get_n();`
- `static public boolean is_clockwise(int a, int b, int c);`
- `static public void give_answer(int s);`

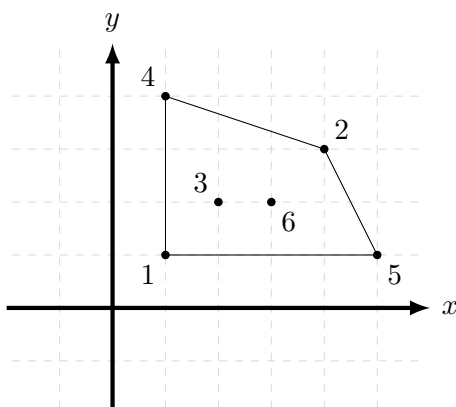
Miután a programod meghívta a `give_answer` függvényt, nem hívhat más függvényeket. A `give_answer` függvényt pontosan egyszer kell hívni.

Ebben a feladatban a programod nem olvashat a standard bemenetről, és nem írhat a standard kimenetre. A `give_answer` hívása után programodnak azonnal terminálnia kell.

Feltételezheted, hogy a pontok koordinátái előre meghatározottak, és nem változnak a program végrehajtása során (vagyis a modul teljes mértékben determinisztikusan viselkedik). Például, ha az alábbi példa tesztelésben a programod végrehajtja a `give_answer(4)` utasítást és rögtön terminál, helyes értékelést kapsz. Megengedett, hogy a program tippeljen a válasz kiszámítása nélkül.

Példa interakció

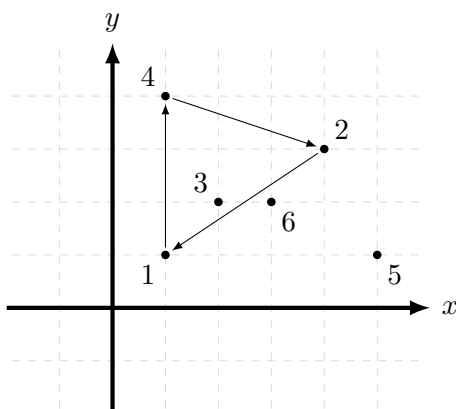
Legyen $n = 6$ város a következő pontokban: $(1, 1)$, $(4, 3)$, $(2, 2)$, $(1, 4)$, $(5, 1)$, $(3, 2)$, amit az alábbi ábra mutat. A konvex burkot vonalakkal jelöltük, amelynek négy csúcsa van, tehát az eredmény 4.



Az alábbi táblázat egy lehetséges interakciót tartalmaz a fenti példához.

Művelet	Kapott érték
<code>get_n()</code>	6
<code>is_clockwise(1, 4, 2)</code>	<code>true</code>
<code>is_clockwise(4, 2, 1)</code>	<code>true</code>
<code>is_clockwise(1, 2, 4)</code>	<code>false</code>
<code>is_clockwise(3, 6, 5)</code>	<code>true</code>
<code>give_answer(4)</code>	-

Az alábbi háromszög az első kérdéshez tartozik. Az 1, 4, 2 pontok által meghatározott háromszög körüljárási iránya az óramutató járásával megegyezik, tehát a kapott érték `true`.



Értékelés

Az alábbi részfeladatok vannak. Minden részfeladat egy vagy több teszt csoportot tartalmaz, és minden teszt csoportban egy vagy több teszteset van.

Minden tesztesetben $3 \leq n \leq 40\,000$. Tesztesetenként az `is_clockwise` függvényt legfeljebb 1 000 000-szor hívhatod meg.

Részfeladat	Korlátok	Pontszám
1	$n \leq 50$	15
2	$n \leq 500$	20
3	$n \leq 15\,000$	20
4	legfeljebb egy pont nem esik a konvex burok oldalaira	20
5	nincs egyéb feltétel	25

Gyakorlás

A `public` mappában kapsz egy példa könyvtári modult, amellyel tesztelheted a programod formális működését. A modul a standard inputról olvassa be az adatokat az alábbi formában:

- Az első sor a városok n számát tartalmazza,
- a következő n sor mindegyike két egész számot tartalmaz, egy város x és y koordinátáját.

A példa modul **nem** ellenőrzi a megoldás helyességét. A bemenet helyességét sem ellenőrzi. Természetesen az értékelő szerver nem ezt a modult használja.

Egy példa bemenet található a `tri0.in` fájlban.

A `give_answer` hívás végrehajtása után a modul kiírja a standard kimenetre az általad adott válasz értékét, és a végrehajtott `is_clockwise` hívások számát.

A megoldásod lefordítására a példa modullal, a következő parancsokat használhatod:

- C: `gcc -O2 -static trilib.c tri.c -lm -std=gnu99`
- C++: `g++ -O2 -static trilib.c tri.cpp -lm -std=c++11`

A megoldásodnak és a modult tartalmazó fájloknak ugyanabban a könyvtárban kell lennie.

Java-ban nincs szükséged semmilyen speciális parancsra a megoldás lefordítására.