Úloha: TRI
Trojúhelníky

czech

CEOI 2018, Den 2. Dostupná paměť: 256 MB. *16.08.2018*

Byteland je krásná země s n ($n \geq 3$) městy, která jsou reprezentována jako n různých bodů v rovině. Města jsou očíslována od 1 do n. Neznáme přesnou polohu jednotlivých měst v Bytelandu. Víme ale, že žádná tři města neleží v jedné přímce.

Konvexním obalem množiny n bodů v rovině nazýváme takový konvexní mnohoúhelník, který má nejmenší možnou plochu a obsahuje všech n zadaných bodů (uvnitř nebo na hranici tohoto mnohoúhelníku). Konvexní mnohoúhelník má všechny vnitřní úhly menší než 180 stupňů a žádné jeho dvě strany se neprotínají.

Vaším úkolem je určit počet vrcholů na hranici mnohoúhelníku, který je konvexním obalem množiny všech měst v Bytelandu. Při řešení tohoto úkolu můžete pokládat otázky na vzájemnou polohu trojic **různých** měst i, j, k ($1 \leq i, j, k \leq n$). Taková otázka se týká trojúhelníku s vrcholy ležícími ve městech i, j, k. Odpověď na tuto otázku nám říká, zda se při průchodu přes vrcholy trojúhelníku v pořadí i, j, k pohybujeme ve směru hodinových ručiček nebo proti směru hodinových ručiček.

Komunikace

Váš program bude používat knihovnu, která vám umožní pokládat otázky a oznámit výsledek výpočtu.

Knihovna (`trilib.h` pro C a C++) obsahuje následující funkce:

- `int get_n();`
  Vrací počet měst.
- `bool is_clockwise(int a, int b, int c);`
  Vrací hodnotu `true`, jestliže vrcholy trojúhelníku a, b, c ($1 \leq a, b, c \leq n, a \neq b \neq c \neq a$) jsou zadány v pořadí po směru hodinových ručiček, a vrací hodnotu `false`, pokud jsou zadány v pořadí proti směru hodinových ručiček.
- `void give_answer(int s);`

Pro jazyk Java obsahuje třída `trilib` následující metody:

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

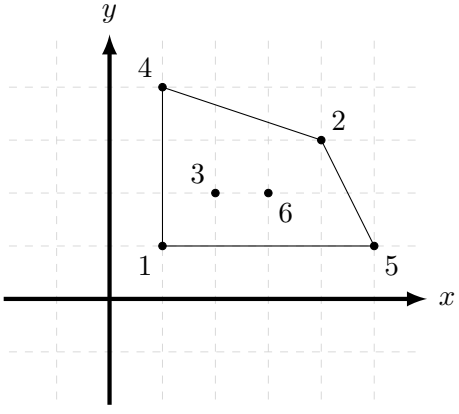
Poté, co váš program zavolá funkci `give_answer`, nesmí už pokládat žádné otázky. Program musí zavolat funkci `give_answer` právě jednou.

V této úloze nesmíte číst žádná data ze standardního vstupu ani zapisovat na standardní výstup. Po zavolání funkce `give_answer`, musí váš program ihned ukončit výpočet.

Můžete předpokládat, že poloha bodů je předem určena a během výpočtu programu se nebude měnit (tzn. knihovna se chová zcela deterministicky). Např. v příkladu uvedeném níže zavolání funkce `give_answer(4)` a následné ukončení výpočtu představuje úspěšné provedení testu. Váš program může zkusit uhodnout správnou odpověď, aniž by ji s jistotou určil.

Příklad výpočtu

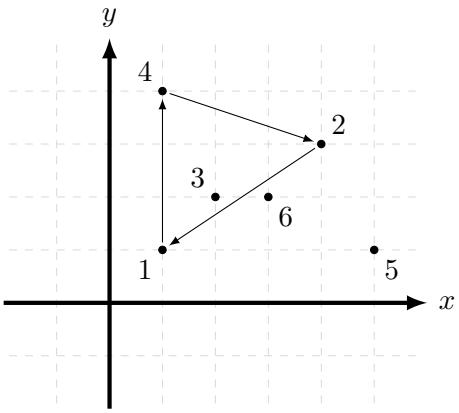
Předpokládejme $n = 6$ měst umístěných na pozicích $(1, 1)$, $(4, 3)$, $(2, 2)$, $(1, 4)$, $(5, 1)$, $(3, 2)$, jak ukazuje obrázek. Konvexní obal je vyznačen úsečkami. Obsahuje čtyři vrcholy na své hranici, takže výsledkem výpočtu je 4.



Následující tabulka ukazuje příklad komunikace s knihovnou, která odpovídá tomuto příkladu.

Volání funkce	Návratová hodnota
`get_n()`	6
`is_clockwise(1, 4, 2)`	`true`
`is_clockwise(4, 2, 1)`	`true`
`is_clockwise(1, 2, 4)`	`false`
`is_clockwise(3, 6, 5)`	`true`
`give_answer(4)`	–

Obrázek ukazuje trojúhelník z prvního dotazu. Města 1, 4, 2 mají pořadí ve směru hodinových ručiček, proto návratová hodnota je `true`.



Hodnocení

Testovací sada je rozdělena do následujících podúloh s dodatečnými omezeními. Testy v každé podúloze jsou seskupeny do jedné či více oddělených skupin testů, z nichž každá může obsahovat jeden či více testů.

Ve všech testech $3 \leq n \leq 40\,000$. V každém testu smíte zavolat funkci `is_clockwise` nejvýše 1 000 000 krát.

Podúloha	Omezení	Body
1	$n \leq 50$	15
2	$n \leq 500$	20
3	$n \leq 15\,000$	20
4	nejvýše jeden bod neleží na hranici konvexního obalu	20
5	žádná další omezení	25

Testování programu

V adresáři `public` najdete ukázkovou knihovnu, která vám umožní otestovat formální správnost vašeho řešení. Tato knihovna čte popis Bytelandu ze standardního vstupu v následujícím formátu:

- na prvním řádku jedno celé číslo n, počet měst,
- na každém z následujících n řádků dvě celá čísla, souřadnice i-tého města.

Tato knihovna **neprovádí** plnou kontrolu vašeho řešení. Nekontroluje ani správnost vstupu. Není to stejná (tajná) knihovna, jako ta na vyhodnocovacím serveru.

Příklad vstupu pro testovací knihovnu najdete v souboru `tri0.in`.

Po zavolání funkce `give_answer` knihovna vypíše na standardní výstup uvedenou odpověď a počet volání funkce `is_clockwise`.

Při překladu vašeho programu s testovací knihovnou můžete použít tyto příkazy:

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

V jazyce Java nemusíte použít žádný zvláštní příkaz.

Soubory s řešením a s knihovnou musí být umístěny ve stejném adresáři.