# Ryby 2. kolo Fiks 2021/2022

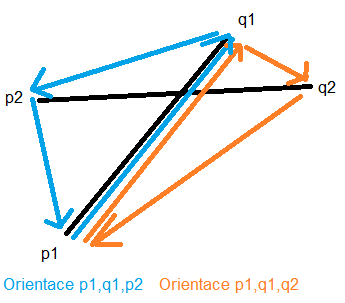
**Kryštof Olík**

Nejprve si zformulujeme zadání bez nepotřebných dodatků. Na vstupu dostaneme několik bodů, které dohromady spojíme úsečkami, a tímto způsobem vytvoříme různé geometrické tvary nazvané polygony. Na konec máme mapu různých polygonů společně s přímkou v určeném směru. Naším úkolem je najít pozici přímky, ve které přímka protíná co nejvíce polygonů a zjistit, kolik polygonů protíná.

První si uložíme směr přímky a zjistíme počet polygonů. Začneme iterovat přes všechny polygony a postupně si je začneme ukládat. Z bodů si uděláme úsečky a z úseček si vytvoříme polygony. Toto ukládání nám zabere O(N) prostoru, kde N je počet bodů. Poté budeme iterovat přes všechny body na mapě a budeme na ně pokládat naší přímku. Tímto způsobem, že vyzkoušíme položit naší přímku na všechny body, máme jistotu, že opravdu najdeme pozici přímky, ve které přímka protíná co nejvíce polygonů.

Naše přímka je ve skutečnosti jenom úsečka, která přesahuje přes celou mapu a dokonce i trochu dál. Začneme iterovat přes úsečky všech polygonů na mapě a následuje náš algoritmus, který kontroluje, zda se naše přímka protíná s úsečkou polygonu.

Nejprve zjistíme všechny čtyři orientace mezi třemi konečnými body na naší úsečce a úsečce polygonu.

Na obrázku vlevo můžeme vidět dvě úsečky nakreslené černou barvou a poté orientaci mezi těmito třemi body. Modrou barvou vidíme, že orientace p1, q1 a p2 je proti směru hodinových ručiček. Zároveň oranžovou barvou vidíme, že orientace p1, q1 a q2 je po směru. Ještě zjistíme další dvě orientace bodů p2, q2, p1 a p2, q2, q1. Také se může stát, že jsou body hned vedle sebe. Tomu říkáme, že je orientace kolineární.

Obecně platí, že pokud jsou první dvě orientace rozdílné a zbylé dvě orientace také rozdílné, tak se úsečky protínají, avšak může nastat situace, kde jedna z orientací je kolineární. V této situaci musíme ověřit, že se určitý bod úsečky nachází na druhé úsečce, a pokud se nachází, tak to znamená, že se protínají.

Aby se polygon považoval za protnutý, tak stačí, aby se naše přímka protnula s jakoukoliv jeho úsečkou, tudíž můžeme zbylé úsečky tohoto polygonu přeskočit, abychom ušetřili trochu času. Připočteme si tedy jedničku do našeho počtu protnutých polygonů a vyzkoušíme, zda se protínáme s dalšími. Tento algoritmus, kde zkoušíme všechna možná místa naší přímky a zkoušíme, zda se protíná se všemi polygony, má časovou složitost O(N2), kde N je počet bodů.

Nakonec máme však ještě dvě důležité otázky: „Funguje náš algoritmus? Je konečný?“ Náš algoritmus najde pokaždé nejvyšší možné množství průseků mezi danou přímkou a všemi polygony, jelikož zkoušíme všechny možné způsoby a společně s naší metodou hledání orientací mezi body máme jistotu, že intersekce detekujeme. Navíc je algoritmus konečný, jelikož jednoduše iteruje pouze přes naše uložené polygony.

K řešení připojuji algoritmus naprogramovaný v jazyce c++. Program samotný je přesným přepisem výše uvedeného algoritmu.