Úkolem je realizovat šablonu třídy, která dokáže vyhledávat cestu mezi dvojicí zadaných míst.

Předpokládáme mapu, ve které se nacházejí místa (uzly) a spojnice mezi uzly (hrany). Pod místem si lze představit města na mapě, adresy počítačů v síti, ... Hrany pak jsou silnice/železnice (pro města), případně datová spojení (pro počítače v síti). Naše třída `CRoute` si bude pamatovat takovou mapu. Protože o uzlech ani hranách nebude nic předpokládat, bude třída realizovaná jako šablona a typ uzlu a typ hrany budou generické parametry této třídy.

Třída bude mít metodu pro přidávání hran a metodu pro vyhledávání. Kompletní rozhraní `CRoute` bude obsahovat:

implicitní konstruktor
: vytvoří prázdnou instanci `CRoute`,

metoda `Add(u1, u2, e)`
: metoda přidá hranu do `CRoute`. Hrana spojuje dvojici uzlů `u1` a `u2`. Parametr `e` udává parametry hrany. Přidávaná hrana je obousměrná.

metoda `Find ( u1, u2[, f] )`

: Metoda nalezne spojení (cestu) mezi dvojicí zadaných uzlů `u1` a `u2`. Výsledkem je seznam (`list`) obsahující uzly tvořící cestu mezi `u1` a `u2`. Prvním prvkem seznamu bude `u1`, posledním `u2`. Spojení mezi uzly nemusí existovat (pak metoda vyhodí výjimku `NoRouteException`), případně spojení může existovat více. Pokud existuje spojení více, bude výsledkem takové spojení, které má nejmenší délku (co nejméně mezilehlých uzlů). Pokud existuje více různých spojení s minimální délkou, pak může být výsledkem libovolné z nich.

  Nepovinným třetím parametrem `f` je filtr na možné použité hrany. Pokud je vynechán, uvažujeme všechny zadané hrany. Pokud je zadán, pak se jedná o funkci/funktor/lambda výraz, který bude aplikován na vlastnost hrany. Hrana bude uvažovaná pouze pokud ji filtr akceptuje (vrátí `true`). Tímto půjde jednoduše volit vlastnosti nalezené cesty, např. budeme chtít pouze cesty, kde hrany umožní zadanou minimální rychlost.

Generický parametr `_T` popisuje uzel. Pro uzly máte garantované následující operace:

- kopírující konstruktor,
- operátor =
- relační operátory (==, !=, < <=, > a >=),
- destruktor,
- operátor pro výpis (<<).
- Další operace s datovým typem nejsou zaručené (mohou, ale nemusí existovat).

Generický parametr `_E` popisuje hranu. Pro hrany máte garantované následující operace:

- kopírující konstruktor,
- operátor =
- destruktor.
- Další operace s datovým typem nejsou zaručené (mohou, ale nemusí existovat).

Odevzdávejte zdrojový kód s implementací šablony třídy `CRoute`. Za základ implementace

použijte přiložený zdrojový kód. Pokud v kódu ponecháte bloky podmíněného překladu, lze takový zdrojový kód lokálně testovat a zároveň jej odevzdávat Progtestu.

Hodnocení je rozděleno na povinnou a nepovinnou část. V povinné části se testují mapy s malým množstvím uzlů. Nepovinný test testuje velké mapy, které vyžadují použití odpovídajících datových struktur. Nezvládnutí nepovinného testu znamená citelnou bodovou ztrátu.

Pro hledání cesty se nejlépe hodí algoritmus BFS (prohledávání do šířky). Pro inspiraci si připomeňte proseminář PA1, ve kterém jsme probírali datové struktury, příklad s cestou v bludišti.

Zdrojový kód s ukázkou práce šablony naleznete v přiloženém archivu.