ZADÁNÍ SEMESTRÁLNÍ PRÁCE
ŘEŠENÍ KOLIZÍ FREKVENCÍ SÍTĚ VYSÍLAČŮ

Zadání

Naprogramujte v ANSI C přenositelnou[1] **konzolovou aplikaci**, která jako vstup načte z parametru příkazové řádky název textového souboru obsahujícího informace o parametrech a pozicích vysílačů na mapě a na jeho základě přidělí každému vysílači frekvenci tak, aby jeho signál nerušil vysílání vysílačů v jeho bezprostředním okolí. Úloha je znázorněna obrázku 1.

Program se bude spouštět příkazem `freq.exe <filename>`. Symbol `<filename>` zastupuje jméno textového souboru, který obsahuje informace o rozmístění vysílačů na mapě a o dostupných vysílacích frekvencích, které jim je možné přidělit.

Váš program tedy může být během testování spuštěn například takto:

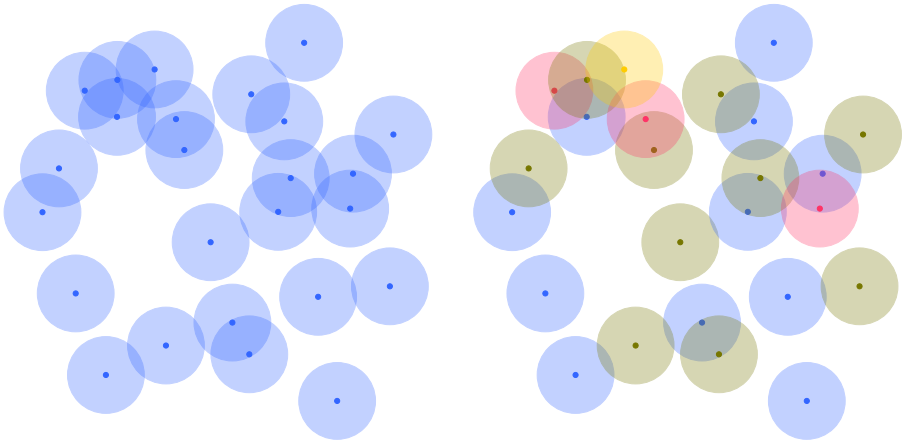
```
...\>freq.exe vysilace-25.txt
```

Výsledkem práce programu bude výpis do konzole, na kterém bude seznam přidělených frekvencí každému vysílači ze vstupního souboru (viz Specifikace výstupu programu). V případě chyby nebo neřešitelné situace nechť program skončí výpisem příslušné chybové hlášky (v angličtině).

Pokud nebude na příkazové řádce uveden právě jeden argument, vypište chybové hlášení a stručný návod k použití programu v angličtině podle běžných zvyklostí (viz např. ukázková semestrální práce na webu předmětu Programování v jazyce C). **Vstupem programu je pouze argument na příkazové řádce – interakce s uživatelem pomocí klávesnice či myši v průběhu práce programu se neočekává.**

Hotovou práci odevzdejte v jediném archivu typu ZIP prostřednictvím automatického odevzdávacího a validačního systému. Archiv nechť obsahuje všechny zdrojové soubory potřebné k přeložení programu, **makefile** pro Windows i Linux (pro překlad v Linuxu připravte soubor pojmenovaný `makefile` a pro Windows `makefile.win`) a dokumentaci ve formátu PDF vytvořenou v typografickém systému TEX, resp. LATEX. Podrobné informace k odevzdání práce najdete na webové stránce předmětu Programování v jazyce C: `https://www.kiv.zcu.cz/studies/predmety/pc/index.php#work` – bude-li některá z částí chybět, kontrolní skript Vaši práci odmítne.

[1] Je třeba, aby bylo možné Váš program přeložit a spustit na PC s operačním prostředím Win32/64 (tj. operační systémy Microsoft Windows NT/2000/XP/Vista/7/8/10) a s běžnými distribucemi Linuxu (např. Ubuntu, Mint, OpenSUSE, Debian, atp.). Server, na který budete Vaši práci odevzdávat a který ji otestuje, má nainstalovaný operační systém Debian GNU/Linux 9.1 Stretch s jádrem verze 3.2.78-1 x86_64 a s překladačem gcc 8.3.0.



Obrázek 1: Znázornění úlohy. Na mapě je vykreslena množina vysílačů (ze vstupního souboru), jejichž pozice je znázorněna tečkou. Vysílač dokáže ale šířit svůj signál pouze do určité vzdálenosti, tj. v kruhu o daném poloměru R. Ve skutečnosti spolu ale signály sousedních vysílačů mohou kolidovat (situace vlevo), protože se kruhy pokrytí signálem překrývají. Každému vysílači je proto potřeba přiřadit jinou frekvenci (barvu) tak, aby signál žádné dvojice vysílačů nekolidoval (vpravo).

Specifikace vstupu programu

Vstupem programu je **pouze** parametr na příkazové řádce: Specifikace souboru (tj. jméno a příp. cesta) s informacemi o vysílačích a frekvencích, které má program přidělovat. Tento soubor je textový a má následující formát:

Available frequencies:
$\mathrm{ID}_{f_0}\ f_0$
$\mathrm{ID}_{f_1}\ f_1$
$\mathrm{ID}_{f_2}\ f_2$
$\vdots$
$\mathrm{ID}_{f_{N-1}}\ f_{N-1}$
Transimission radius:
R
Transmitters:
$\mathrm{ID}_{t_0}\ X_{t_0}\ Y_{t_0}$
$\mathrm{ID}_{t_1}\ X_{t_1}\ Y_{t_1}$
$\mathrm{ID}_{t_2}\ X_{t_2}\ Y_{t_2}$
$\vdots$
$\mathrm{ID}_{t_{M-1}}\ X_{t_{M-1}}\ Y_{t_{M-1}}$

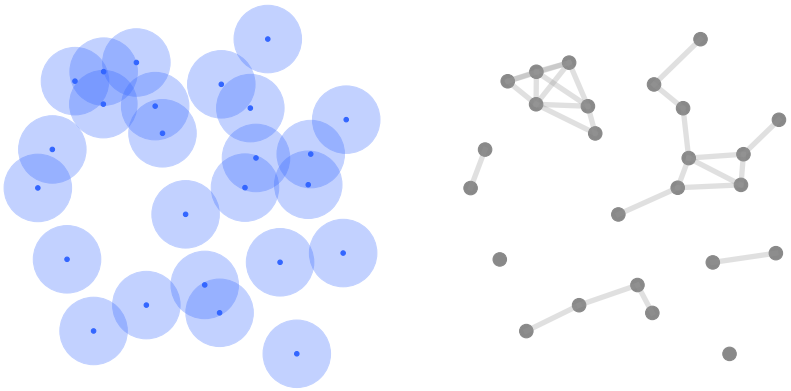
```
Available frequencies:
0 93400
1 104600
2 139700
3 154600
Transmission radius:
15
Transmitters:
0 115.698096 3.112792
1 95.047235 112.320582
2 74.776052 33.719497
3 29.709430 114.079607
4 25.366625 13.250972
5 102.803973 143.009002
6 0.592240 76.828840
7 121.893144 91.878910
8 108.263298 43.781410
9 137.666118 107.186368
10 81.381655 21.325507
11 56.001114 101.120042
12 66.274976 65.102099
13 92.665047 76.970736
14 97.559577 90.155843
15 120.783480 78.247073
16 136.297332 47.885413
17 13.568902 45.105008
18 17.097654 124.302199
19 7.034448 93.943072
```

ID_{f_i} je číselný identifikátor i-té vysílací frekvence a f_i je příslušná vysílací frekvence v Hertzích. N je celkový počet dostupných vysílacích frekvencí a M je celkový počet vysílačů. Konstanta R udává poloměr kruhu v kilometrech, v němž vysílač poskytuje pokrytí signálem a kde může také docházet ke kolizím s vysílači pracujícími na stejné frekvenci.

V sekci `Transmitters` se nachází identifikátory a pozice jednotlivých vysílačů. ID_{t_i} je identi-

fikační číslo i-tého vysílače a X_{t_i} a Y_{t_i} jsou jeho příslušné kartézské souřadnice (X, Y) na *ploché* mapě (relativně k O(x, y), tj. nemají žádný zeměpisný význam). Tyto souřadnice jsou stejně jako R udávány v kilometrech.

Poznámka: Můžete předpokládat, že veškerá identifikační čísla jsou vždy uvedena vzestupně od nuly a jsou ve vstupním souboru seřazena.



Obrázek 2: Převod informací o vysílačích na graf kolizí. Pokud se signály (kruhy) překrývají, může dojít ke kolizi, a proto jsou tyto vysílače (vrcholy) v grafu spojeny hranou.

Graf kolizí

K vyřešení úlohy je třeba sestrojit graf kolizí, který bude mít tyto vlastnosti:

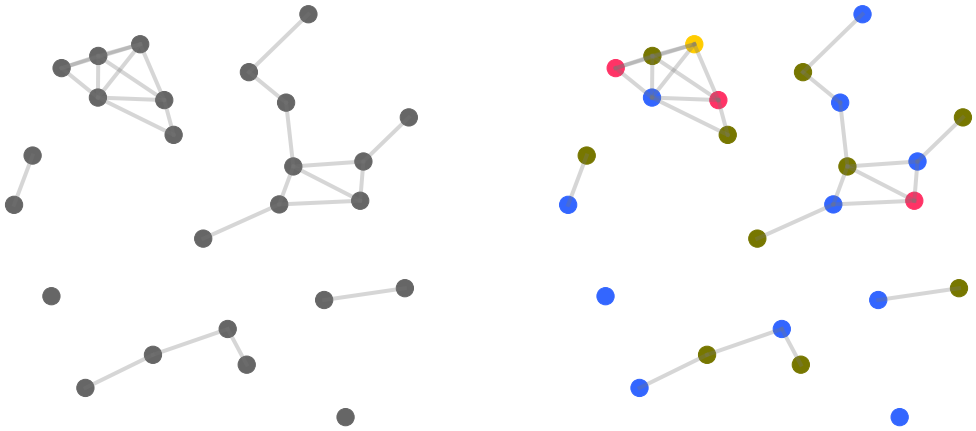
- Každý vysílač je v grafu reprezentován unikátním vrcholem.
- Vrcholy (vysílače) propojuje hrana pouze tehdy, pokud mezi nimi může docházet ke kolizi signálu (tzn. jejich oblasti pokrytí signálem se překrývají).
- Každý vrchol nese informaci o přiřazené frekvenci. Tato informace je vyjádřena jako celé číslo, které se v teorii grafů označuje jako *barva* vrcholu. Nechť barva vrcholu odpovídá identifikačním číslům frekvencí ID_{f_i} ve vstupním souboru.

Na obrázku 2 je znázorněn převod informací o vysílačích na graf kolizí podle uvedených pravidel. Jako postačující podmínku kolize uvažujte situaci, kdy je eukleidovská vzdálenost mezi dvěma vysílači menší než $2R$ (dvojnásobek poloměru kruhu, který vysílač pokrývá signálem).

Obarvování grafu

Úlohu přidělování frekvencí vysílačů lze řešit technikou z oblasti teorie grafů, která se nazývá *obarvování grafu*. Obarvování grafu je postup, při kterém každému vrcholu přiřadíme celé číslo (barvu) tak, aby každá dvojice vrcholů spojená hranou měla rozdílnou barvu. Zároveň požadujeme, aby k tomuto obarvení bylo použito co nejméně barev (frekvencí vysílačů). V tomto případě bude obarvován graf kolizí. Rozdílná barva zde znamená rozdílnou vysílací frekvenci, a je tedy zřejmé, že pokud správně obarvíme kolizní graf, ke kolizi signálů nemůže dojít. Správně obarvený graf je znázorněn na obrázku 3.

Úloha správného obarvení grafu s použitím nejmenšího počtu barev je ve skutečnosti tzv. *NP-úplný problém*, což znamená, že dosud není známý algoritmus, který by problém dokázal optimálně vyřešit s polynomiální časovou složitostí. Existuje ale algoritmus, který problém řeší v mnoha případech optimálně nebo je jím poskytované řešení blízké optimálnímu.



Obrázek 3: Neobarvený graf kolizí (vlevo) a jeho správně obarvená varianta (vpravo). V obarveném grafu žádná dvojice vrcholů spojená hranou nemá shodnou barvu (a tedy i příslušné vysílače mají různou frekvenci a nekolidují spolu.)

Popis tohoto algoritmu následuje:

1. Vytvoř graf kolizí.
2. Nastav všechny vrcholy grafu jako dosud neobarvené.
3. Procházej všechny vrcholy grafu a vlož do zásobníku první neobarvený vrchol.
4. Není-li zásobník prázdný, vyjmi vrchol ze zásobníku.
5. Vybranému vrcholu postupně přiřazuj barvy $0, 1, ..., N-1$, dokud není nalezena barva, která nekoliduje se sousedními vrcholy (těmi, do kterých z aktuálního vrcholu vede hrana). Přiřaď aktuálnímu vrcholu nejnižší možnou hodnotu barvy, která tuto podmínku splňuje a vlož do zásobníku všechny dosud neobarvené sousední vrcholy právě zpracovaného vrcholu. Pokud je nejnižší nalezená hodnota barva větší než největší dostupná hodnota barvy (čili největší použitelná hodnota frekvence, $N-1$), algoritmus ukonči a označ úlohu jako neřešitelnou.

   (*Poznámka:* Úloha sice řešení mít může, ale uvedený algoritmus jej nedokázal nalézt.)
6. Opakuj postup od bodu 4. Je-li zásobník prázdný, pokračuj na další neobarvený vrchol v grafu (bod 3).

Popsaný algoritmus rovněž pěkně ilustruje video na adrese:
`http://www.youtube.com/watch?v=o3B-V94VnQg`

Řešitelnost úlohy

Tato úloha **není vždy řešitelná** – existují varianty vstupu, pro které nelze graf kolizí obarvit tak, aby byl použit pouze zadaný počet frekvencí. Dojde-li k tomu, že program nebude schopen úlohu vyřešit, nechť zareaguje chybovým hlášením podle níže uvedené ukázky (s přesným dodržením formátování):

```
ERR#3: Non-existent solution!
```

Upozornění: Graf, který bude Váš program obarvovat, může mít i tisíce vrcholů a hran, a je proto nezbytně nutné zvolit vhodný způsob implementace vnitřní reprezentace grafu tak, aby výpočet proběhl v rozumném čase.

Specifikace výstupu programu

Výstup programu bude směřován pouze na konzoli. Výstupem (v případě, že nenastala chyba) bude vzestupně seřazená posloupnost identifikačních čísel vysílačů a jim přiřazená vysílací frekvence v Hertzích, například:

```
0␣92300
1␣105600
2␣92300
3␣86200
4␣86200
```

Každá dvojice těchto hodnot je vypsána na samostatné řádce a ukončena znakem konec řádky '\n' – nic jiného program vypisovat **nesmí**, jinak neprojde validačním skriptem. Pokud algoritmus řešení nenašel, nechť program vypíše chybu č. 3 (viz výše a níže).

Pro testovací účely jsou připojeny vstupní soubory s příslušnými vzorovými výstupy. Graf lze samozřejmě obarvit mnoha různými způsoby tak, aby byla splněna podmínka korektního obarvení; berte proto výstup pouze jako příklad jednoho možného správného řešení.

V souboru `vysilace-25.txt` a `vysilace-25-vysledek.txt` se nachází testovací množina 25-ti vysílačů, resp. odpovídající výstup pro tento vstupní soubor.

Podobně v souborech `vysilace-1000.txt` a `vysilace-1000-vysledek.txt` se nachází vstupní soubor pro 1000 vysílačů a jemu odpovídající vzorové obarvení (tato množina je vhodná pro testování rychlosti vašeho programu).

Chybové stavy

Dojde-li při vykonávání programu k chybě, nechť reaguje výpisem jedné z chybových hlášek uvedených v tabulce. Mějte, prosím, na paměti, že z důvodu automatické kontroly odevzdávaného díla je nezbytně nutné **přesně dodržet** formát chybových hlášek.

Chybové hlášení	Význam, popis chybového stavu
`ERR#1:␣Missing argument!`	Na příkazové řádce nebyl program předán parametr specifikující vstupní textový soubor s informacemi o vysílačích.
`ERR#2:␣Out of memory!`	Není k dispozici dostatek operační paměti.
`ERR#3:␣Non-existent solution!`	Řešení pro daný počet vysílacích frekvencí nebylo nalezeno.

Hodnoty chyb od 4 výše můžete využít pro vlastní potřeby, ovšem dodržte uvedené formátování, tj. velkými písmeny zkratka `ERR`, následovaná bez mezer znakem '`#`' a číslem chyby – za číslem budiž pak dvojtečka a mezera a pak stručný popis chyby ukončený vykřičníkem ('`!`').

Číselný kód chyby nechť program předá také operačnímu systému prostřednictvím návratové hodnoty z funkce `int main(···)`. V případě, že program proběhne bez chyby, nechť je návratová hodnota programu 0.

Užitečné techniky a odkazy

Uvedené techniky je možné využít při řešení úlohy. Protože se jedná o postupy víceméně standardní, lze k nim na Internetu nalézt velké množství dokumentace:

1. teorie grafů – `https://en.wikipedia.org/wiki/Graph_theory`
2. barvení grafu – `http://en.wikipedia.org/wiki/Graph_coloring`
3. zásobník

Řešení úlohy je zcela ve vaší kompetenci – zvolte takové algoritmy a techniky, které podle vás nejlépe povedou k cíli.