# 1. DOMÁCÍ ÚKOL

Lineární programování LS 2018/19 — Odevzdat do úterý 30. dubna 2019 8:00

## Úkoly obecně

Úkoly odevzdávejte emailem vašemu cvičícímu a předmět emailu začněte `LP: HW1`. Vypracovaný úkol se skládá z generátoru LP/IP a dokumentace. Na adrese `https://iuuk.mff.cuni.cz/~husek/foo/opt-1819-ukol-1.zip` naleznete archiv s připravenými vstupy, na nichž máte váš program testovat.[1] Kromě základních vstupů označených `vstup`$i$`-`$x$`.txt`, kde $i$ je číslo úlohy a $x$ je číslo vstupu, (a k těmto vstupům se vztahuje doba běhu uvedená níže), archiv dále obsahuje malé vstupy označené `-s`, (pravděpodobně) problematické vstupy pro první úlohu označené `-c` a soubory `reseni1.txt` a `reseni2.txt`, které obsahují pro každý vstup řádek se jménem vstupního souboru, optimální hodnotou účelové funkce a časem běhu v sekundách vzorového řešení na testovacím stroji.[2]

Generátor LP/IP je program v rozumném programovacím jazyce, který transformuje vstup (formát vstupu viz jednotlivé úlohy) na LP/IP program v jazyce GNU MathProg pro řešič glpsol. Jednoduchý manuál ke GNU MathProg naleznete na `https://iuuk.mff.cuni.cz/~bohm/texts/mathprog_intro_cz.html`, oficiální dokumentace je součástí distribuce glpk, případně rozumně aktuální verze je online na `https://ktiml.mff.cuni.cz/~micka/let1718/optimalizace/gmpl.pdf`.

Vstup načítejte ze standardního vstupu a výstup vypisujte na standardní výstup (pro potřeby debugovacích hlášek používejte standardní chybový výstup). Můžete předpokládat, že vstup je vždy ve validním formátu. Formát výstupu vygenerovaného LP/IP (tj. co vypíše `glpsol -m vygenerovane_lp.mod`) je popsán u jednotlivých příkladů, v dokumentaci popište, co znamená, pokud vaše LP/IP nemá řešení (v tomto případě LP/IP nemusí vypisovat povinnou část výstupu).

Odevzdávejte zdrojový kód, ne přeložený program. Také si dejte pozor, aby k vašemu řešení nebyly přibaleny nadbytečné soubory jako nepořádek vygenerovaný Visual Studiem (ani soubor popisující projekt Visual Studia nepotřebujeme) či složku `__MACOS_X__`.

Za rozumné určitě považujeme jazyky C, C++, Java, C#, Python, Perl, Bash. Zdrojový kód musí být možné spustit či zkompilovat v počítačové laboratoři Rotunda na (libovolném) počítači s Linuxem. Můžete používat standardní knihovny příslušného jazyka (jsou-li k dispozici v labu). Pokud si nejste jistí, jestli se daná knihovna dá považovat za standardní, zeptejte se cvičího.

Odevzdaný zdrojový kód generátoru by měl být čitelný, formátovaný a v rozumné míře okomentovaný. Na druhou stranu, vygenerovaný lineární program může být nečitelný a dlouhý, jak jen potřebujete.

Nedílnou součástí řešení je dokumentace, která musí obsahovat:

1. popis, jak program sestavit (nejlépe pokud stačí `make`),
2. informace, jak program ovládat,
3. stručný popis, jak bude vypadat vaše LP/IP, a
4. co znamená, když vygenerovaný LP/IP nemá řešení.

Dokumentaci odevzdejte ve formátu pdf nebo plain text (Markdown je povolený). Dokumentace nemusí být dlouhá; měla by se vejít na 1 nebo 2 stránky.

**Upozorňujeme**, že řešení, která (kombinatoricky) vyřeší zadanou úlohu, a pak na ní pustí triviální LP, nebudou uznávána. Cílem tohoto domácího úkolu je seznámit se s tvorbou lineárních programů. Slibujeme, že druhý domácí úkol bude větší výzva, která jen tak snadno kombinatoricky nepůjde.

Je-li cokoli nejasné, zeptejte se svého cvičícího.

[1] Váš generátor samozřejmě musí fungovat i pro jiné podobně velké vstupy.

[2] Testovací stroj má procesor Intel Celeron 3865U (1.80GHz, Kaby Lake), a tedy není příliš rychlý.

# Úloha 1 – Topologické uspořádání [10 bodů]

Na vstupu máte orientovaný graf. Vaším úkolem je napsat LP, který nalezne (částečné) uspořádání vrcholů takové, že pro každou hranu $uv$ platí, že vrchol $u$ je v daném uspořádání menší než $v$. Uspořádání modelujte pomocí nezáporných celých čísel a snažte se minimalizovat největší použitou hodnotu. Očekávaná doba běhu na připravených vstupech je cca 10 sekund.

## Formát vstupu

Soubor s orientovaným grafem má následující formát: První řádek začíná slovem `DIGRAPH` a za ním následuje počet vrcholů a počet hran, obé odděleno mezerami, a na konci prvního řádku je dvojtečka. Vrcholy jsou číslovány od nuly. Další řádky mají tvar $i$ `-->` $j$ a určují jednotlivé hrany. Příklad $K_{1,2}$:

```
DIGRAPH 3 2:
 0 --> 1
 0 --> 2
```

## Formát výstupu

Program může vypisovat jakékoli informace uznáte za vhodné, ale výstup vždy musí obsahovat následující povinnou část: Povinná část je ohraničena řádky `#OUTPUT:` $M$ a `#OUTPUT END`, kde $M$ je maximum z čísel přiřazených vrcholům. Mezi nimi je výpis pořadí vrcholů ve tvaru `v_`$i$`:` $x$, kde $i$ je číslo vrcholu a $x$ jeho pořadí. Pořadí musí být nezáporné celé číslo. Příklad pro orientaci $K_{1,2}$ uvedenou výše:

```
#OUTPUT: 1
v_0: 0
v_1: 1
v_2: 1
#OUTPUT END
```

# Úloha 2 – Graf bez krátkých cyklů [15 bodů]

Na vstupu máte orientovaný graf s váhovou funkcí na hranách. Vaším úkolem je napsat IP, který najde nejmenší váženou podmnožinu hran takovou, že po jejím odebrání graf nebude obsahovat žádnou **orientovanou** kružnici délky 4 či kratší. Graf na vstupu neobsahuje smyčky ani cykly délky 2. Očekávaná doba běhu na připravených vstupech je 5 až 120 sekund dle vstupu.

## Formát vstupu

Soubor s orientovaným grafem má následující formát: První řádek začíná slovem `WEIGHTED DIGRAPH` a za ním následuje počet vrcholů a počet hran, obé odděleno mezerami, a na konci prvního řádku je dvojtečka. Vrcholy jsou číslovány od nuly. Další řádky mají tvar $i$ `-->` $j$ `(`$w$`)` a určují jednotlivé hrany, $w$ je nezáporná celočíselná váha hrany. Příklad $K_4$:

```
WEIGHTED DIGRAPH 4 6:
 0 --> 1 (4)
 0 --> 2 (3)
 0 --> 3 (1)
 1 --> 2 (4)
 2 --> 3 (2)
 3 --> 1 (5)
```

## Formát výstupu

Program může vypisovat jakékoli informace uznáte za vhodné, ale výstup vždy musí obsahovat následující povinnou část: Povinná část je ohraničena řádky `#OUTPUT:` $W$ a `#OUTPUT END`, $W$ je celková váha odebraných hran. Mezi nimi je výpis odebraných hran ve tvaru $i$ `-->` $j$. Příklad pro vstup výše:

```
#OUTPUT: 2
2 --> 3
#OUTPUT END
```