# Úvod do TI 2018 - Úloha 3

## Všeobecné poznámky

1. Nezabúdajte na papier napísať deň, kedy chodíte na cvičenie alebo meno cvičiacej, aby sa k vám jednoduchšie dostali úlohy.
2. Ak máte jeden príklad na viac papierov, zopnite ich, nech sa nám nestratia a neprídete kvôli tomu o body.
3. Iba výsledok nestačí. Treba písať aj zdôvodnenia. Ak nájdete niečo na internete, uveďte aspoň zdroj, ideálne sa to snažte dokázať a vysvetliť.

## Poznámky k príkladu 3.1

(opravovala Danka Pardubská)
Najväčším problémom viacerých vás je, že nie celkom rozumiete tomu, čo je to dôkaz. Vyhýbate sa slovám ako nech, potom,...a snažíte sa písať len matematické značky a postupnosti znakov tak, ako ste to asi videli na tabuli. Nepochopením ale vznikajú neplatné tvrdenia. Nie je jednoduché písať dôkaz, kvôli tomu tie úlohy sú. Skúste argumentovať slovensky tak, ako by ste to vysvetľovali kolegovi a k zápisu tej slovenskej argumentácie používajte vhodne matematický formalizmus. neviem to lepšie vysvetliť':-)

Ku koncu som už asi nemala silu písať každému to isté, preto možno komentáre nie sú rovnaké.

**Bodovanie** - 1a, 1c boli za 4 body (dve implikácie), 1b bolo za dva body.

Keï som bola presvedčená, že myslené to bolo dobre ale zapísané zle (písali ste nepresne len $\forall x$), strhla som za to body len v jednom z 1a, 1c - pre povzbudenie:-)

**Príklad 1a:**

- Nestačí napísať "vyplýva z definície homomorfizmu"bez toho, aby ste ju čo i len napísali. Nie na domácej úlohe, ktorá je zameraná na to, aby ste ukázali, že viete korektne formálne dokázať jednoduché vlastnosti o jazykoch a ich vlastnostiach.
- Dôkaz $h(L_1)h(L_2) = h(L_1L_2)$ začína:

  $\Rightarrow \forall x_1 \in (h(L_1)h(L_2) = h(L_1L_2)$

  $\Rightarrow \forall x_1, x_2, \exists y_1, y_2\ h(y_1)h(y_2) = x_1x_2 \wedge y_1 \in L_1 \wedge y_2 \in L_2$

  Dobre myslené, zle nekorektne napísané. Ako to malo byť?

  Nech $x \in h(L_1)h(L_2)$. Potom $\exists y_1 \in L_1, y_2 \in L_2 : x = h(y_1)h(y_2)$, prípadne, ak to ďalej treba, pokračujete: Nech $x_1 = h(y_1), x_2 = h(y_2)$

Ak stále dávate implikácie a kvantifikátory, strácate spojenie medzi symbolmi, ktoré máte namysli. Lebo vy chcete hovoriť o $x \in h(L_1)h(L_2)$ ale ten predpoklad nezopakujete a píšete $\forall x$

- $x \in h(L_1)h(L_2) \Rightarrow x = ab, a \in h(L_1), b \in h(L_2)$ je možno presnejšie (len aby ste si to uvedomili) $x \in h(L_1)h(L_2) \Rightarrow \exists a \in h(L_1), b \in h(L_2)$ také, že $x = ab$

**Príklad 1b:** Tento príklad skoro každý zvládol.

- Niekto nevidel rozdiel medzi $\{\emptyset\}$ a $\emptyset$. Pozor, $\{\emptyset\}$ obsahuje jeden prvok, prázdnu množinu.
- Niekto vyrobil spor tým, že definoval extra $h(00)$ a $h(0)$, pričom $h(00) \neq h(0)h(0)$, teda mu to nespĺňalo podmienku homomorfizmu.

**Príklad 1c:**

- Viacerí ste použili v argumentácii medzikrok

$$\exists y((y \in L_1 \wedge x \in h^{-1}(y)) \wedge (y \in L_2 \wedge x \in h^{-1}(y))) \Leftrightarrow$$

$$\exists y(y \in L_1 \wedge x \in h^{-1}(y)) \wedge \exists y(y \in L_2 \wedge x \in h^{-1}(y))$$

  pri ktorom pochybujem, že ste si uvedomili, že vo všeobecnosti

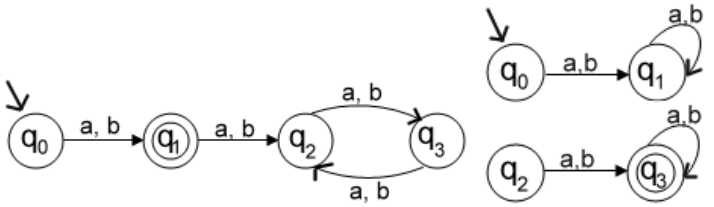
$$\exists y(f(y) \wedge g(y)) \neq \exists y\ f(y) \wedge \exists y\ g(y)$$

- $h(w) \in \Sigma^*$, preto $h(w)$ je slovo a nie množina. Nemôžme teda pre slová $v, w$ písať $v \in h(w)$
- $\forall x \in h^{-1}(L_1) \cap h^{-1}(L_2) \overset{niejepravda}{\Rightarrow} \forall x \exists y\ h^{-1}(y) = x \wedge y \in L_1$
  Tvrdenie malo byť nejako takto:
  Nech $x \in h^{-1}(L_1) \cap h^{-1}(L_2)$. Potom $\exists y :\ x = h^{-1}(y) \wedge y \in L_1 ...$

  Tým "nech $x \in h^{-1}(L_1) \cap h^{-1}(L_2)$" sme jasne vymedzili, aké $x$ máme v ďalšej argumentácii na mysli. Ak použijete následne $\forall x$ bez toho, aby ste o $x$ povedali viac, všeobecný kvantifikátor prebil vlastnosť, ktorú ste chceli $x \in h^{-1}(L_1) \cap h^{-1}(L_2)$.
- častejšie sa objavilo: $\forall x \in h^{-1}(L_1) \cap h^{-1}(L_2) \Rightarrow v \cap w | v \in h^{-1}(L_1) \cap w \in h^{-1}(L_2)$
  to asi niekto niekde zle napísal a viacerí ste to prebrali, nie je mi celkom jasné, čo to chcelo byť; o.i. ste operáciu prieniku nemali definovanú na slovách

## Poznámky k príkladu 3.2

(opravovala: Daniela Bezáková)
Príklad bol za 10 bodov. Bodovanie bolo nasledovné: a) 2 body, b) 3 body, c) 2 body, d) 3 body.
a), b) a c) boli prevažne bez chýb. Za drobné chyby som strhla zväčša len nejaké desatiny bodu, max 0,5. d) Za správnu odpoveď, že automat rozpoznáva nekonečný jazyk, som dávala 1 bod, za správne zdôvodnenie 2 body. Najčastejšie nedostatočné odpovede boli :




(a) (b)

Obr. 1

- Automat obsahuje cyklus. To, že automat obsahuje nejaký cyklus, ešte neznamená, že rozpoznáva nekonečný jazyk. Napr. automat na obr. 1a) cyklus obsahuje, ale akceptuje iba slová $a, b$. Za takéto zdôvodnenie bolo -1,5b.
- Automat obsahuje cykly, z ktorých sa dá dostať do akceptačného stavu. Toto je už takmer správna formulácia, ale nikto neuvažoval o tom, či sa do tých cyklov, (z ktorých sa dá dostať do akceptačného stavu,) vieme dostať zo vstupného stavu. Napríklad automat n obr. 1b) obsahuje cyklus, z ktorého sa dá dostať do akceptačného stavu, ale rozpoznáva prázdny jazyk. Za toto zdôvodnenie bolo -0,5b.
- V odpovedi ste spomínali nekonečne dlhé slovo. Slovo nemôže byť nekonečne dlhé. Napr. slovo $a^i$, je (konečné) slovo dĺžky $i$, bez ohľadu na to, aké veľké je $i$. Až keď spravíme zjednotenie takých slov $a^i$ pre všetky $i \in N$, dostaneme nekonečný jazyk. Zdôvodnenie, že automat rozpoznáva nekonečne dlhé slovo, bolo za 0 bodov (pokiaľ tam okrem toho nebolo ešte niečo iné).
- K iným úvahám nájdete poznámky priamo vo Vašej úlohe.

## poznámky k príkladu 3.3

(m.w)
Príklad za 10b.
Použili ste rekurziu alebo cyklus. Niektorí ste použili `replace`, čo bolo celkom vtipné.
Najčastejšou chybou bolo, že ste slovo neprepisovali, ale vypočítali ste koľko konkrétnych znakov má byť vo výsledku (7b), drobnou chybou bolo malé nedodržanie obrázku, zvyčajne ste mali o dĺžku o jednu viac alebo menej (-0.5b)
Ak ste riešili úplne inú úlohu, dostali ste 1b
Neodpustím si ešte poznámku ku konštrukciám typu:

```
if w == 'F':
   result += 'FF'
if w == 'f':
   result += 'F+f-f-f+F'
if w == '+':
   result += '+'
if w == '-':
   result += '-'
```

na to slúži konštrukcia `if ... elif...elif ...`
Generovanie slova sa dalo napísať napríklad takto:

```
def slovo(n):
  vys = 'FfF'
  for i in range(n):
    vys = vys.replace('F', 'FF').replace('f', '+FfF-FfF-FfF+')
  return vys
```