# Cvičenie 7 - Bezkontextové gramatiky, zásobníkové automaty

Ing. Viliam Hromada, PhD.

C-510
Ústav informatiky a matematiky
FEI STU

viliam.hromada@stuba.sk

18.11.2020

# Príklad č. 1

Je daná bezkontextová gramatika $G_1 = (V, T, P, S)$, $V = \{S\}$, $T = \{a, b\}$. $S$ je počiatočný neterminál. Pravidlá gramatiky:

1. $S \rightarrow aSb$
2. $S \rightarrow \varepsilon$

Úlohy:

1. Zistite, či je jazyk popísaný gramatikou prázdny.
2. Zistite, či reťazce: *ab*, *aabb*, *aaab*, *bbaa* majú v gramatike odvodenie.
3. Popíšte, aký jazyk gramatika generuje.

Ako zistiť, či je jazyk generovaný gramatikou prázdny:

1. Pre každý symbol gramatiky zistíme, či je tzv. **generujúci symbol**, t.j. či pre symbol $X$ platí, že $X \Rightarrow^* w$, kde $w \in T^*$. Inými slovami, či pre symbol $X$ gramatiky (symbol je aj terminál, aj neterminál) platí, že sa z neho dá na aspoň 0 krokov odvodiť nejaký reťazec terminálov.
2. Ak zistíme, že počiatočný neterminál $S$ je generujúci symbol, potom sa z neho určite dajú odvodiť nejaké reťazce a jazyk generovaný gramatikou **nemôže byť prázdny**.
3. Ak počiatočný neterminál $S$ **nie je generujúci symbol**, potom je jazyk generovaný gramatikou prázdny.

Ako zistíme, ktoré symboly sú generujúce:

1. Každý terminálny symbol gramatiky je generujúci, lebo pre $a \in T$ platí $a \Rightarrow^* a$.
2. Ak je v gramatike pravidlo $A \rightarrow \alpha$, kde $A \in N, \alpha \in (V \cup T)^*$, t.j. $A$ je neterminál a $\alpha$ je reťazec terminálov a neterminálov, a zároveň vieme, že **každý symbol** $\alpha$ je **generujúci symbol**, tak aj $A$ musí byť **generujúci symbol**.
3. Rovnako, ak je v gramatike pravidlo $A \rightarrow \varepsilon$, tak $A$ je tiež **generujúci symbol**.
4. Postupne teda analyzujeme symboly a pravidlá gramatiky, kým nenájdeme všetky generujúce symboly.

V našej gramatike:

1. $S \rightarrow aSb$
2. $S \rightarrow \varepsilon$

sú generujúce symboly:

1. $a$, $b$ sú generujúce symboly určite, lebo sú to terminály.
2. Keďže v gramatike je pravidlo $S \rightarrow \varepsilon$, tak aj $S$ je generujúci symbol.
3. Tým sme vyčerpali všetky symboly - všetky sú generujúce.
4. Keďže aj počiatočný neterminál **je generujúci symbol**, tak jazyk generovaný gramatikou **nie je prázdny**.

Zistite, či *ab* má v gramatike odvodenie:

- Hľadáme odvodenie reťazca *ab* v gramatike, t.j. zisťujeme, či v gramatike platí, či $S \Rightarrow^* ab$
- Teda, či vieme začať s počiatočným neterminálom $S$ a postupnou aplikáciou pravidiel z neho *odvodiť (derivovať)* daný reťazec.

Keď začneme s počiatočným neterminálom máme 2 možnosti:

1. $S \Rightarrow \varepsilon$ ,t.j. použijeme pravidlo č. 2. Dostali sme $\varepsilon$, z ktorého derivácia nevie pokračovať ďalej. Tadiaľto cesta nevedie...
2. $S \Rightarrow aSb$ , t.j. použijeme pravidlo č. 1. Dostali sme reťazec $aSb$, v ktorom môžeme znovu v ďalšom kroku použiť niektoré pravidlo na neterminál $S$.
   - 2.1 $S \Rightarrow aSb \Rightarrow aaSbb$, t.j. na neterminál $S$ v $aSb$ sme použili pravidlo č. 1. Tým dostávame reťazec $aaSbb$, čiže vo výsledku by boli na začiatku $aa$. Ale náš reťazec nezačína $aa$, ale len $a$. Preto toto nie je dobrá cesta
   - 2.2 $S \Rightarrow aSb \Rightarrow ab$, t.j. na neterminál $S$ v $aSb$ sme použili pravidlo č. 2. Dostávame **práve** $ab$, čiže naša úloha je úspešne ukončená.

Zistite, či *aabb* má v gramatike odvodenie:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aa\varepsilon bb = aabb$$

Teda reťazec *aabb* patrí do jazyka generovaného gramatikou $G_1$. Toto vieme popísať aj $aabb \in L(G_1)$.

Zistite, či $aaab$ má v gramatike odvodenie:

1. Vidíme, že v reťazec začína 3-krát písmenom $a$.
2. Keďže jediné pravidlo, ktoré vie vytvoriť v reťazci písmeno $a$ je $S \to aSb$, musíme toto pravidlo 3-krát použiť.
3. Avšak, toto pravidlo zároveň 3-krát vyrobí písmeno $b$, viď:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb$$

4. V našom reťazci $aaab$ je však len 1-krát písmeno $b$.
5. Preto vidíme, že v danej gramatike sa $aaab$ odvodiť nedá!
6. Platí teda $aaab \notin L(G_1)$.

Zistite, či *bbaa* má v gramatike odvodenie:

1. Vidíme, že v reťazec začína 2-krát písmenom $b$.
2. Keďže jediné pravidlo, ktoré vie vytvoriť v reťazci písmeno $b$ je $S \to aSb$, museli by sme toto pravidlo 2-krát použiť.
3. Avšak, toto pravidlo zároveň 3-krát vyrobí písmeno $a$, ktoré navyše **vždy** bude stáť pred písmenom $b$, resp. na začiatku odvádzaného reťazca, viď:

$$S \Rightarrow aSb \Rightarrow aaSbb$$

4. V našom reťazci je však prvý symbol písmeno $b$
5. Preto vidíme, že v danej gramatike sa *bbaa* odvodiť nedá!
6. Platí teda $bbaa \notin L(G_1)$.

Popíšte, aký jazyk gramatika generuje:

1. Jazyk generovaný gramatikou predstavujú všetky reťazce terminálov, ktoré sa v nej dajú odvodiť z počiatočného neterminálu na nejaký počet krokov, t.j.

$$L(G) = \{w \mid S \Rightarrow^* w, w \in T^*\}$$

2. Aby sme zistili, aký jazyk gramatika generuje, sa musíme zamyslieť, **aké všetky reťazce** sa dajú pomocou nej odvodiť, resp. **akú majú nejakú spoločnú vlastnosť**, čo pre ne platí:
3. Vidíme, že sa dá odvodiť prázdny reťazec $S \Rightarrow \varepsilon$.
4. Toto pravidlo sa zároveň použije **vždy** na ukončenie derivácie.
5. Pretože druhé pravidlo $S \rightarrow aSb$ vždy síce vyrobí 2 terminály $a$ a $b$, avšak aj nový neterminál $S$, preto toto pravidlo **nikdy** nemôže deriváciu ukončiť.
6. Čo sa stane, ak toto pravidlo aplikujeme $n$-krát?

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow ... \Rightarrow a^n Sb^n$$

7. Ak teraz deriváciu ukončíme pomocou $S \rightarrow \varepsilon$, dostaneme:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow ... \Rightarrow a^n Sb^n \Rightarrow a^n b^n$$

1. Teda vidíme, že gramatika generuje **všetky reťazce obsahujúce rovnaký počet** $a$ **a** $b$**, pričom zároveň sú v reťazci najprv všetky symboly** $a$ **a potom všetky symboly** $b$
2. Teda jazyk $L(G_1) = \{a^n b^n \mid n \geq 0\}$.
3. U uvedenom zápise je zachytený aj fakt, že $\varepsilon \in L(G_1)$, pretože $a^0 b^0 = \varepsilon$.

# Príklad č. 2

Je daná bezkontextová gramatika $G_2 = (V, T, P, S)$, $V = \{S, A\}$, $T_1 = \{a, b\}$. $S$ je počiatočný neterminál. Pravidlá gramatiky:

1. $S \rightarrow aA$
2. $A \rightarrow aA$
3. $A \rightarrow bA$

Úlohy:

1. Zistite, či je jazyk popísaný gramatikou prázdny.
2. Zistite, či reťazce: *ab*, *aabb*, *aaab*, *bbaa* majú v gramatike odvodenie.
3. Popíšte, aký jazyk gramatika generuje.

Najprv zistíme, či je jazyk generovaný gramatikou, $L(G_2)$ prázdny, t.j. či $L(G_2) = \emptyset$. Generujúce symboly gramatiky sú:

1. Terminály $a, b$, pretože $a \Rightarrow^* a$, resp. $b \Rightarrow^* b$.
2. Zatiaľ teda vieme, že generujúce symboly sú určite $\{a, b\}$.
3. V gramatike nie je pravidlo v tvare $S \to \varepsilon$ alebo $A \to \varepsilon$, teda týmto spôsobom o $S$ alebo $A$ nevieme povedať, či sú generujúce.
4. Rovnako platí, že vo všetkých pravidlách je na pravej strane symbol, o ktorom nevieme, či je generujúci.
5. Preto usudzujeme, že ani $S$, ani $A$ **nie sú generujúce symboly**.
6. A keďže **počiatočný neterminál** nie je generujúci symbol, platí, že jazyk generovaný gramatikou **je prázdny**, a teda gramatika **negeneruje žiadne reťazce.**
7. $L(G_2) = \emptyset$

O tom sa viete presvedčiť aj tak, že ak by ste skúšali vyrobiť deriváciu ľubovoľného reťazca $w$ z terminálov:

$$S \Rightarrow \ldots \Rightarrow w$$

tak by sa vám to nepodarilo, lebo by vám **vždy** zostával 1 neterminál $A$. Teda derivácia sa **nikdy nedá ukončiť**.

Preto úlohy, či reťazce *ab*, *aabb*, *aaab*, *bbaa* majú v gramatike odvodenie riešiť **nemusíme**, alebo vieme, že odvodenie mať **nemôžu.**

Rovnako popis jazyka, ktorý gramatika generuje, je veľmi jednoduchá:

$$L(G_2) = \emptyset$$

teda gramatika $G_2$ generuje prázdny jazyk $\emptyset$.

# Príklad č. 3

Je daná bezkontextová gramatika $G_3 = (V, T, P, S)$, $V = \{S, A\}$, $T_1 = \{a, b, c\}$. $S$ je počiatočný neterminál. Pravidlá gramatiky:

1. $S \to aA$
2. $A \to aA$
3. $A \to bA$
4. $A \to c$

Úlohy:

1. Zistite, či je jazyk popísaný gramatikou prázdny.
2. Zistite, či reťazce: *abc*, *abac*, *aba*, *babc* majú v gramatike odvodenie.
3. Popíšte, aký jazyk gramatika generuje.

Najprv zistíme, či je jazyk generovaný gramatikou, $L(G_3)$ prázdny, t.j. či $L(G_3) = \emptyset$. Generujúce symboly gramatiky sú:

1. Terminály $a, b, c$, pretože sú to terminály :).
2. Zatiaľ teda vieme, že generujúce symboly sú určite $\{a, b, c\}$.
3. V gramatike nie je pravidlo v tvare $S \to \varepsilon$ alebo $A \to \varepsilon$, teda týmto spôsobom o $S$ alebo $A$ nevieme povedať, či sú generujúce.
4. Pre neterminál $S$ máme len pravidlo $S \to aA$ a o $A$ zatiaľ nevieme, či je generujúci, čiže ani o $S$ nevieme, či je generujúci.
5. Pre neterminál $A$ však existuje pravidlo $A \to c$, teda pravidlo, v ktorom sú všetky symboly na pravej strane určite generujúce.
6. Preto usudzujeme, že aj $A$ je generujúci symbol, teda generujúce sú určite $\{A, a, b, c\}$.

Pokračujeme:

1. Keď sa teraz znovu vrátime k pravidlu $S \to aA$, na pravej strane už o oboch symboloch vieme, že sú generujúce, teda aj neterminál $S$ je generujúci symbol.
2. Vieme teda, že generujúce symboly sú $\{S, A, a, b, c\}$. Keďže sú to všetky symboly gramatiky, môžeme končiť.
3. Keďže počiatočný neterminál $S$ **patrí medzi generujúce symboly**, gramatika bude určite vedieť **generovať nejaký reťazec**, a teda $L(G_3) \neq \emptyset$.

Zistite, či $abc$ má v gramatike odvodenie:

1. Na začiatku máme pre počiatočný neterminál len jednu možnosť:
$$S \Rightarrow aA$$
2. toto pravidlo vyrobí na začiatku výsledného reťazca $a$. Zatiaľ teda ok.
3. Teraz máme pre neterminál $A$ 3 možnosti, ktoré pravidlo použiť (pravidlo č. 2, 3 alebo 4):
$$S \Rightarrow aA \Rightarrow aaA$$
$$S \Rightarrow aA \Rightarrow abA$$
$$S \Rightarrow aA \Rightarrow ac$$
4. Prvá možnosť je derivácia, ktorá ešte nekončí, ale na začiatku slova vyrobí predponu $aa$. Náš reťazec však **nemá predponu** $aa$. Preto tento krok je zlý!
5. Tretia možnosť je derivácia, ktorá zároveň aj skončila a derivovala reťazec $ac$. Keďže $ac \neq abc$, táto možnosť je tiež zlá.
6. Druhá možnosť je derivácia, ktorá ešte neskončila, ale vyrába reťazec, ktorý začína predponou $ab$. Vidíme, že toto by mohla byť cesta k úspešnému výsledku!

Zistite, či *abc* má v gramatike odvodenie (pokračovanie):

1. Zatiaľ sme odvodili reťazec:
$$S \Rightarrow^* abA$$
2. Znovu máme pre neterminál $A$ 3 možnosti, ktoré pravidlo použiť (pravidlo č. 2, 3 alebo 4):
$$S \Rightarrow^* abA \Rightarrow abaA$$
$$S \Rightarrow^* abA \Rightarrow abbA$$
$$S \Rightarrow^* abA \Rightarrow abc$$
3. Vidíme, že tretia možnosť nám dáva **práve** deriváciu reťazca *abc*, teda sme zistili, že daný reťazec **má v gramatike nasledovné odvodenie**
$$S \Rightarrow aA \Rightarrow abA \Rightarrow abc$$
4. A teda platí, že $abc \in L(G_3)$.

Zistite, či *abac* má v gramatike odvodenie: Toto je len variácia predchádzajúceho výpočtu, platí, že áno, lebo:

$$S \Rightarrow aA \Rightarrow abA \Rightarrow abaA \Rightarrow abac$$

. Teda aj $abac \in L(G_3)$.

Zistite, či *aba* má v gramatike odvodenie:

1. Ak by sme chceli odvodiť reťazec *aba* musíme najprv vyrobiť predponu *ab* - to vieme v derivácii urobiť:
$$S \Rightarrow aA \Rightarrow abA$$
2. Teraz by sme potrebovali z neterminálu $A$ urobiť terminál $a$. Avšak pravidlá nám dovolia alebo urobiť z $A \to c$, t.j.
$$S \Rightarrow aA \Rightarrow abA \Rightarrow abc$$
3. to však **nie je reťazec** *aba*! Respektíve vieme síce odvodiť predponu *aba*:
$$S \Rightarrow aA \Rightarrow abA \Rightarrow abaA$$
4. ale stále nám tam ostáva neterminál $A$, ktorého sa nevieme zbaviť **bez toho**, aby nám tam nevznikol ďalší terminál!
5. Preto reťazec *aba* sa v gramatike odvodiť **nedá!** A teda $aba \notin L(G_3)$.

Zistite, či *aba* má v gramatike odvodenie:

1. Ekvivalentne si stačí uvedomiť, že jediný spôsob, ako ukončiť deriváciu je, že sa ako posledné pravidlo aplikuje $A \to c$
2. A z tvaru pravidiel vyplýva, že neterminál $A$ **vždy na konci** reťazca symbolov gramatiky počas derivácie.
3. Teda **každý odvoditeľný reťazec** musí **končiť** symbolom $c$.
4. A keďže reťazec *aba* **nekončí symbolom** $c$, tak **nemôže** patriť do jazyka generovaného gramatikou.

Okrem toho, že každý reťazec, ktorý má v gramatike odvodenie, musí končiť terminálom $c$, platí ešte jedna vlastnosť:

1. Všimnite si, že **prvé pravidlo**, ktoré sa vždy dá použiť, je $S \rightarrow aA$.
2. Teda **každý reťazec**, ktorý má v gramatike deriváciu, bude **vždy začínať** terminálom $a$.
3. Preto, ak by sme chceli zistiť, či má v gramatike deriváciu reťazec $babc$, tak vidíme, že určite **nie!**
4. Pretože neexistuje spôsob, ako urobiť deriváciu reťazca začínajúceho písmenom $b$ z počiatočného neterminálu!
5. Preto $babc \notin L(G_3)$.

Popíšte, aký jazyk gramatika generuje.

1. Pohľadom na gramatiku vidíme, že prvý symbol každého generovaného reťazca musí byť $a$, lebo pre počiatočný neterminál máme jediné pravidlo $S \to aA$.
2. Čiže prvý symbol bude $a$ a za ním bude všetko to, čo vieme odvodiť z neterminálu $A$. Pre ten máme pravidlá:
   - $A \to aA$
   - $A \to bA$
   - $A \to c$
3. Čiže všetko, čo odvodíme z neterminálu $A$ končí symbolom $c$. Navyše z neho vieme postupne odvádzať reťazce, ktoré obsahujú ľubovoľnú postupnosť z množiny $\{a, b\}$.
4. Z neterminálu $A$ teda vieme dostávať reťazce v tvare $\{a, b\}^*c$.
5. A z počiatočného neterminálu $S$ reťazce v tvare $a\{a, b\}^*c$.
6. Jazyk generovaný gramatikou je teda jazyk popísateľný ako $a\{a, b\}^*c$, alebo dokonca aj regulárnym výrazom $a(a + b)^*c$.

- To, že v tomto prípade je jazyk generovaný bezkontextovou gramatikou popísateľný regulárnym výrazom, a teda že sa jedná nielen o bezkontextový, ale aj o regulárny jazyk, je dané tým, že regulárne jazyky tvoria podmnožinu bezkontextových jazykov.
- Nie vždy sa to však dá, napríklad v prvom príklade bol jazyk $a^n b^n$ a ten sa **nedá** popísať regulárnym výrazom.

# Príklad č. 4

Je daná bezkontextová gramatika $G_4 = (V, T, P, S)$, $V = \{S, A, B\}$, $T_1 = \{a, b, c\}$.
$S$ je počiatočný neterminál. Pravidlá gramatiky:

- $S \rightarrow AcB \mid BA$
- $A \rightarrow aAb \mid a$
- $B \rightarrow bAa$

Úlohy:

1. Zistite, či je jazyk popísaný gramatikou prázdny.
2. Popíšte, aký jazyk gramatika generuje.

Najprv zistíme, či je jazyk generovaný gramatikou, $L(G_4)$ prázdny, t.j. či $L(G_4) = \emptyset$. Generujúce symboly gramatiky sú:

1. Terminály $a, b, c$.
2. Zatiaľ teda vieme, že generujúce symboly sú určite $\{a, b, c\}$.
3. V gramatike nie je pravidlo v tvare $S \to \varepsilon$, $A \to \varepsilon$ alebo $B \to \varepsilon$, teda týmto spôsobom o $S, A, B$ nevieme povedať, či sú generujúce.
4. Pre neterminál $A$ však existuje pravidlo $A \to a$, teda pravidlo, v ktorom sú všetky symboly na pravej strane určite generujúce.
5. Preto usudzujeme, že aj $A$ je generujúci symbol, teda generujúce sú určite $\{A, a, b, c\}$.
6. Po tomto zistení vieme povedať, že aj $B$ je generujúci symbol, pretože má pravidlo $B \to bAa$,v ktorom sú na pravej strane všetky symboly také, že o nich vieme, že sú určite generujúce. Teda generujúce symboly sú určite $\{A, B, a, b, c\}$.

(pokračovanie)

7 A už vieme povedať aj to, že $S$ je generujúci symbol, pretože má pravidlo $S \rightarrow AcB$,v ktorom sú na pravej strane všetky symboly také, že o nich vieme, že sú určite generujúce. Teda generujúce symboly sú určite $\{S, A, B, a, b, c\}$.

7 Ekvivalentne by sme to vedeli určiť aj podľa pravidla $S \rightarrow BA$, v ktorom taktiež vieme, že na pravej strane sú len symboly, o ktorých vieme, že sú určite generujúce.

8 A keďže $S$ je generujúci symbol, tak jazyk $L(G_4)$ určite nie je prázdny.

## Popíšte, aký jazyk gramatika generuje.

Pohľadom na pravidlá sa pokúsime určiť, v akom tvare sú reťazce, ktoré gramatika generuje. Keďže všetky začínajú od počiatočného neterminálu a preň existujú dve pravidlá:

1. $S \to AcB$
2. $S \to BA$

tak vidíme, že reťazce, ktoré gramatika generuje budú v nasledovných tvaroch:

1. budú v tvare $w_1 c w_2$, kde $w_1$ je predpona, ktorá bude generovaná z neterminálu $A$, za ňou je symbol $c$ a za ním je prípona $w_2$ generovaná z neterminálu $B$, alebo
2. budú v tvare $w_1 w_2$, kde $w_1$ je predpona generovaná z neterminálu $B$, za ktorou je prípona $w_2$ generovaná z neterminálu $A$.

Preto sa potrebujeme bližšie pozrieť na neterminály $A$, $B$, aké reťazce sa dajú generovať z nich.

Pre neterminál $A$ máme 2 pravidlá:

1. $A \to aAb$
2. $A \to a$

Z pravidiel vidíme, že:

1. Z $A$ vieme odvodiť reťazec začínajúci $a$, končiaci $b$ medzi ktorými bude znovu reťazec odvoditeľný z $A$
2. Z $A$ vieme odvodiť jedno $a$-čko.
3. To znamená, že prvé pravidlo vieme počas derivácie rekurzívne opakovať a deriváciu ukončíme druhým pravidlom. Čiže reťazce by mohli byť v nasledovnom tvare:

$$A \Rightarrow aAb \Rightarrow^* a^n A b^n \Rightarrow a^n a b^n = a^{n+1} b^n$$

Teda z neterminálu $A$ vieme odvádzať reťazce v tvare $a^{n+1} b^n$, kde $n \geq 0$.

Pre neterminál $B$ máme 1 pravidlo:

1. $B \to bAa$

Z pravidla vidíme, že:

1. Z $B$ vieme odvodiť reťazec začínajúci $b$, končiaci $a$ medzi ktorými bude reťazec odvoditeľný z neterminálu $A$.
2. Z predchádzajúcej analýzy vieme, že z neterminálu $A$ vieme odvodiť reťazce tvaru $a^{n+1}b^n$, kde $n \geq 0$.
3. Preto z neterminálu $B$ vieme odvodiť:

$$B \Rightarrow bAa \Rightarrow^* ba^{n+1}b^na$$

Teda z neterminálu $B$ vieme odvádzať reťazce v tvare $ba^{n+1}b^na$, kde $n \geq 0$.

Keď teraz vieme, že z neterminálov:

1. $A$ vieme odvodiť reťazce tvaru $a^{n+1}b^n$, kde $n \geq 0$
2. $B$ vieme odvodiť reťazce tvaru $ba^{n+1}b^n a$, kde $n \geq 0$

tak keď sa vrátime k počiatočnému neterminálu

1. $S \rightarrow AcB$
2. $S \rightarrow BA$

vidíme, že vieme odvodiť reťazce tvarov:

1. $S \Rightarrow AcB \Rightarrow^* a^{n+1}b^n cba^{m+1}b^m a$, kde $n \geq 0$, $m \geq 0$
2. $S \Rightarrow BA \Rightarrow ba^{n+1}b^n aa^{m+1}b^m$, kde $n \geq 0$, $m \geq 0$.

Výsledný jazyk $L(G_4)$ je teda zjednotením 2 menších jazykov:

$$L(G_4) = \{a^{n+1}b^ncba^{m+1}b^ma \mid n \geq 0, m \geq 0\} \cup \{ba^{n+1}b^naa^{m+1}b^m \mid n \geq 0, m \geq 0\}$$

POZOR!!! Jazyk:

$$L(G_4) = \{a^{n+1}b^ncba^{n+1}b^na \mid n \geq 0\} \cup \{ba^{n+1}b^naa^{n+1}b^n \mid n \geq 0\}$$

by **nebol správna odpoveď** pretože je potrebné odlíšiť to, že časti pochádzajúce z neterminálov $A, B$ sú **nezávislé!** Preto je potrebné použiť viacero premenných $n, m$, ktoré označujú počet odvodených terminálov.

# Nájdite bezkontextovú gramatiku

Nájdite bezkontextové gramatiky k jazykom:

1. $L_1 = \{w \in \{a, b\}^* \mid \sharp_a(w) = \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a, b$ rovnaký.
2. $L_2 = \{w \in \{a, b\}^* \mid \sharp_a(w) > \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a$ väčší než počet symbolov $b$.
3. $L_3 = \{w \in \{a, b\}^* \mid \sharp_a(w) < \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a$ menší než počet symbolov $b$.
4. $L_4 = \{wcw^R \mid w \in \{a, b\}^*\}$, t.j. palindrómy v strede ktorých je symbol $c$.
5. $L_5 = \{xaby \mid x \in \{a, b\}^*, y \in \{a, b\}^*\}$, t.j. reťazce z písmen $\{a, b\}$ obsahujúce $ab$ ako podreťazec
6. $L_6 = \{a^n b^n c^m \mid n \geq 0, m \geq 0\}$

# Príklad č. 1

Nájdite bezkontextovú gramatiku k jazyku: $L_1 = \{w \in \{a,b\}^* \mid \sharp_a(w) = \sharp_b(w)\}$, t.j. reťazce z písmen $\{a,b\}$, v ktorých je počet symbolov $a$, $b$ rovnaký.

Potrebujeme nájsť takú gramatiku $G$, aby $L(G) = L_1$. To znamená, že z počiatočného neterminálu $S$ budeme vedieť odvodiť taký reťazec, v ktorom je počet symbolov $a$ a $b$ rovnaký, t.j. napríklad:

- $\varepsilon$ (0-krát $a$, aj $b$)
- $ab$, $ba$ (1-krát $a$, 1-krát $b$)
- $aabb$, $abab$, $abba$, $baab$, $baba$, $bbaa$ (2-krát aj $a$, aj $b$)
- $aaabbb$, $aabbba$, $aabbab$, ... (3-krát aj $a$, aj $b$)
- ...

Skúsme sa zamyslieť, ako vyzerajú reťazce, ktoré obsahujú rovnaký počet $a$, aj $b$ z hľadiska ich konštrukcie:

1. Najkratší možný reťazec je $\varepsilon$.
2. Ak by sme už mali reťazec, ktorý obsahuje rovnaký počet $a$ a $b$ a chceli by sme zaň pridať ďalšie $a$, musíme zabezpečiť, aby sme mali istotu, že sa zároveň niekedy neskôr vygeneruje aj $b$.
3. Rovnako, ak by sme k reťazcu, ktorý spĺňa rovnosť počtu $a$ a $b$ chceli pridať $b$, musíme zabezpečiť, aby sa neskôr pridalo aj $a$.

Keď $S$ je neterminál, z ktorého vieme odvodiť reťazce s rovnakým počtom $a$ a $b$, tak z prvého bodu vyplýva pravidlo $S \to \varepsilon$.

Rekurzívne vytváranie reťazca spôsobom, že máme reťazec s korektným počtom $a$ a $b$ a pridáme zaň symbol $a$ a zároveň si chceme zaistiť to, že v budúcnosti sa vygeneruje aj $b$, zapíšeme pomocou pravidla

$$S \to SaB$$

kde neterminál $B$ signalizuje, že *chýba terminál b*. Preto z neho môžeme vyvodiť 2 pravidlá:

- alebo priamo vytvorí terminál $b$, t.j. $B \to b$
- alebo, ak by sme chceli znovu vygenerovať $a$, tak nám vlastne už budú chýbať $b$-čka dve, t.j. $B \to aBB$

Analogická situácia vznikne, ak máme reťazec s korektným počtom $a$ a $b$ a chceme zaň pridať symbol $b$ a zároveň si chceme zaistiť to, že v budúcnosti sa vygeneruje aj $a$:

$$S \rightarrow SbA$$

kde neterminál $A$ signalizuje, že *chýba terminál a*. Preto z neho môžeme vyvodiť 2 pravidlá:

- alebo priamo vytvorí terminál $a$, t.j. $A \rightarrow a$
- alebo, ak by sme chceli znovu vygenerovať $b$, tak nám vlastne už budú chýbať $a$-čka dve, t.j. $A \rightarrow bAA$

Keď si uvedené zhrnieme do jednej gramatiky, tak dostaneme gramatiku s pravidlami:

- $S \to \varepsilon \mid SaB \mid SbA$
- $A \to a \mid bAA$
- $B \to b \mid aBB$

a s neterminálmi $V = \{S, A, B\}$ a terminálmi $T = \{a, b\}$, kde $S$ je počiatočný neterminál.

Je zrejmé, že uvedená gramatika $G$ určite generuje reťazce, kde je počet $a$ a $b$ rovnaký, t.j. $L(G) \subseteq L_1$.

Taktiež každý reťazec, ktorý obsahuje rovnaký počet symbolov $a$ a $b$ v gramatike odvodenie, t.j. $L_1 \subseteq L(G)$.

Teda $L(G) = L_1$.

Bez ďalšieho vysvetľovania uvádzame ešte iné gramatiky, ktoré **rovnako dobre** generujú všetky reťazce s rovnakým počtom *a* a *b*:

- $S \rightarrow \varepsilon \mid aBS \mid bAS$
- $A \rightarrow a \mid bAA$
- $B \rightarrow b \mid aBB$

(alebo gramatika len s 1 neterminálom *S*)

- $S \rightarrow \varepsilon \mid SaSbS \mid SbSaS$

(alebo iná gramatika len s 1 neterminálom *S*)

- $S \rightarrow \varepsilon \mid aSb \mid bSa \mid SS$

# Príklad č. 2

Nájdite bezkontextovú gramatiku k jazyku: $L_2 = \{w \in \{a, b\}^* \mid \sharp_a(w) > \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a$ väčší než počet symbolov $b$.

Potrebujeme nájsť takú gramatiku $G$, aby $L(G) = L_2$. V predchádzajúcom príklade sme našli takú, kde je počet symbolov $a$ rovnaký ako počet symbolov $b$. Jej jednoduchou úpravou môžeme nájsť gramatiku pre tento jazyk.

Táto gramatika generuje reťazce s rovnakým počtom $a$ a $b$

- $S \rightarrow \varepsilon \mid SaB \mid SbA$
- $A \rightarrow a \mid bAA$
- $B \rightarrow b \mid aBB$

Aby sme dostali reťazce, kde je viac symbolov $a$ než $b$, tak potrebujeme do gramatiky doplniť len generovanie symbolov $a$. Najjednoduchšie vyzerá byť to, že do pravidiel pre neterminál $S$ doplníme pravidlo:

$$S \rightarrow Sa$$

Toto pravidlo umožní na ľubovoľnom mieste v derivovanom reťazci vygenerovať "nadbytočné" $a$-čka, t.j. docielime, aby ich počet bol väčší, než $b$-čok.

Dostávame teda gramatiku

- $S \rightarrow \varepsilon \mid SaB \mid SbA \mid Sa$
- $A \rightarrow a \mid bAA$
- $B \rightarrow b \mid aBB$

Určite teda budeme vedieť generovať také reťazce, kde je viac *a*-čok než *b*-čok. Avšak naša úprava nie je úplne dobrá, pretože v tejto gramatike stále vieme generovať aj $\varepsilon$, alebo *abab*, čo **však nesmieme!**, lebo v nich **nie je** počet *a* väčší než počet *b*.

Keďže najkratší reťazec, ktorý by mal patriť do jazyka už nie je $\varepsilon$, ale *a*, tak vymeníme pravidlo $S \rightarrow \varepsilon$ za pravidlo $S \rightarrow a$.

Výsledná gramatika

- $S \to a \mid SaB \mid SbA \mid Sa$
- $A \to a \mid bAA$
- $B \to b \mid aBB$

Teraz už korektne generuje práve také reťazce, kde je počet $a$ väčší než počet $b$. Určite existujú aj iné gramatiky, ktoré by to vedeli. Táto mala tú výhodu, že sme ju vyrobili len malou úpravou z gramatiky pre reťazce, kde je počet $a$ a $b$ rovnaký.

# Príklad č. 3

Nájdite bezkontextovú gramatiku k jazyku: $L_3 = \{w \in \{a, b\}^* \mid \sharp_a(w) < \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a$ menší než počet symbolov $b$.

Táto gramatika je len drobnou úpravou gramatiky z predchádzajúcej úlohy. Bez vysvetľovania:

- $S \to b \mid SaB \mid SbA \mid Sb$
- $A \to a \mid bAA$
- $B \to b \mid aBB$

# Príklad č. 4

Nájdite bezkontextovú gramatiku k jazyku: $L_4 = \{wcw^R \mid w \in \{a, b\}^*\}$, t.j. palindrómy v strede ktorých je symbol $c$.

Tento jazyk tvoria reťazce:

- $c$ (najkratší možný reťazec)
- $aca$, $bcb$ (vľavo a vpravo od $c$ je reťazec dĺžky 1)
- $aacaa$, $abcba$, $bacab$, $bbcbb$ (vľavo a vpravo od $c$ je reťazec dĺžky 2)
- ...

Pre konštrukciu gramatiky pre tento jazyk je dôležité si uvedomiť, že:

1. Najkratší reťazec, ktorý do jazyka patrí je $c$.
2. Ak $w$ je reťazec, ktorý patrí do jazyka, t.j. je to palindróm, ktorého stredný symbol je $c$, tak potom aj $awa$, aj $bwb$ budú tiež palindrómy, ktorých stredný symbol je $c$.

Keďže neterminál $S$ je ten, z ktorého majú byť reťazce odvodené, tak v podstate $S$ reprezentuje palindrómy so stredným symbolom $c$. Na základe 2 uvedených vlastností teda vieme odvodiť nasledovné pravidlá:

1. $S \to c$
2. $S \to aSa$
3. $S \to bSb$

Výsledná gramatika pre jazyk $L_4$ je teda $G = (V, T, P, S)$, kde neterminály $V = \{S\}$, terminály $T = \{a, b, c\}$, počiatočný neterminál je $S$ a pravidlá:

1. $S \to c$
2. $S \to aSa$
3. $S \to bSb$

Z tvaru pravidiel gramatiky je zrejmé, že všetko, čo pomocou nich vygenerujeme, bude palindróm s prostredným symbolom $c$, t.j. že $L(G) \subseteq L_4$.

Taktiež, ak máme daný ľubovoľný palindróm so stredným symbolom $c$, tak určite bude mať v gramatike odvodenie - keďže stačí postupne aplikovať pravidlá č. 2 a 3 podľa toho, aké symboly sa v reťazci nachádzajú pred stredným symbolom, napr. ak $w = abcba$, tak potom

$$S \Rightarrow aSa \Rightarrow abSba \Rightarrow abcba$$

Teda aj $L_4 \subseteq L(G)$.

A teda $L_4 = L(G)$, t.j. naša gramatika je korektná.

# Príklad č. 5

Nájdite bezkontextovú gramatiku k jazyku: $L_5 = \{xaby \mid x \in \{a, b\}^*, y \in \{a, b\}^*\}$, t.j. reťazce z písmen $\{a, b\}$ obsahujúce *ab* ako podreťazec.

Vidíme, že reťazce jazyka sa skladajú z 3 častí:

1. Prefixu $x$, ktorý predstavujú ľubovoľné reťazce z písmen $\{a, b\}$
2. Pevne daného podreťazca *ab*
3. Sufixu $y$, ktorý predstavujú ľubovovoľné reťazce z písmen $\{a, b\}$

Keďže každý generovaný reťazec by mal pozostávať z týchto 3 častí, tak si to vieme zapísať pomocou pravidla:

$$S \rightarrow ABC$$

kde

- $A$ bude neterminál, z ktorého odvodíme prefix $x$
- $B$ neterminál, z ktorého odvodíme $ab$
- $C$ neterminál, z ktorého odvodíme sufix $y$.

Ak chceme pomocou gramatiky odvodiť *ľubovoľnú postupnosť daných symbolov*, v našom prípade $\{a, b\}$, tak sa to jednoducho dá urobiť nasledovne:

$$A \to aA \mid bA \mid \varepsilon$$

T.j. rekurzívnym opakovaním pravidiel vieme vytvoriť ľubovoľný reťazec z $\{a, b\}$ a deriváciu ukončíme posledným nerekurzívnym pravidlom $A \to \varepsilon$.

Výsledná gramatika $G = (V, T, P, S)$ by teda mohla mať pravidlá:

- $S \to ABC$
- $A \to aA \mid bA \mid \varepsilon$
- $B \to ab$
- $C \to aC \mid bC \mid \varepsilon$

kde neterminály $V = \{S, A, B, C\}$, terminály $T = \{a, b\}$ a $S$ je počiatočný neterminál.

Znovu platí, že všetko, čo gramatika generuje patrí do jazyka $L_5$, t.j. $L(G) \subseteq L_5$ a zároveň, že všetko, čo patrí do jazyka $L_5$ má v gramatike odvodenie, t.j. $L_5 \subseteq L(G)$.

Preto $L(G) = L_5$

# Príklad č. 6

Nájdite bezkontextovú gramatiku k jazyku: $L_6 = \{a^n b^n c^m \mid n \geq 0, m \geq 0\}$.

Reťazce z tohto jazyka sa teda akoby skladajú z 2 častí:

1. Na začiatku majú symboly $a$ a $b$, ktorých je rovnaký počet ($n$) a sú usporiadané v štýle "najprv $a$, potom $b$.
2. Za $b$-čkami sa nachádza postupnosť symbolov $c$, ktorých počet je **nezávislý** od počtu $a$, respektíve $b$.

To, že sa reťazce skladajú z 2 častí, ktoré sú nezávislé, znovu môžeme zapísať napríklad:

$$S \to AB$$

kde:

- Z neterminálu $A$ odvodíme tú časť, kde je rovnaký počet $a$ a $b$ usporiadaný v tvare $a ... ab ... b$
- Z neterminálu $B$ odvodíme potrebný počet $c$ symbolov

Odvodenie reťazcov z neterminálu $A$ (t.j. v podstate reťazce v tvare $a^n b^n$):

$$A \rightarrow aAb \mid \varepsilon$$

Odvodenie reťazcov z neterminálu $B$ (t.j. v podstate reťazce $c^m$, čo je to isté ako $c^*$)

$$B \rightarrow cB \mid \varepsilon$$

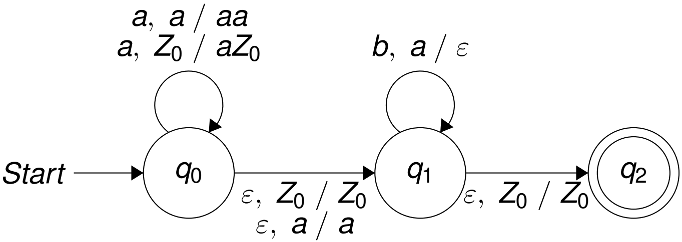
Výsledná gramatika $G = (V, T, P, S)$ by teda mohla mať pravidlá:

- $S \to AB$
- $A \to aAb \mid \varepsilon$
- $B \to cB \mid \varepsilon$

kde neterminály $V = \{S, A, B\}$, terminály $T = \{a, b, c\}$ a $S$ je počiatočný neterminál. Znovu je (dúfam) ľahké vidieť, že naozaj $L(G) = L_6$.

# Príklad č. 1

Je daný zásobníkový automat $P_1 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, ktorého prechodová funkcia je daná obrázkom:




Úlohy:

1. Zistite, či ZA akceptuje reťazce $\varepsilon, ab, ba$
2. Popíšte, aký jazyk akceptuje ZA.

Pre úplnosť popis ZA:

- Stavy zásobníkového automatu $Q = \{q_0, q_1, q_2\}$
- Abeceda vstupných symbolov $\Sigma = \{a, b\}$
- Abeceda zásobníkových symbolov $\Gamma = \{Z_0, a\}$
- Počiatočný stav: $q_0$
- Počiatočný zásobníkový symbol $Z_0$
- Akceptačné stavy: $F = \{q_2\}$

Aby sme zistili, či je nejaký reťazec akceptovaný ZA, musíme simulovať jeho výpočet. Simuláciu zapisujeme pomocou konfigurácií, ktoré nám udávajú:

- V akom stave sa ZA nachádza
- Aká je neprečítaná časť vstupu
- Aký je obsah zásobníka

Napríklad, ak chceme spracovať vstup $ab$, tak na začiatku je automat v stave $q_0$ a v zásobníku je len počiatočný zásobníkový symbol $Z_0$. Teda počiatočná konfigurácia ZA by bola $(q_0, ab, Z_0)$.

Prechodová funkcia udáva, čo sa udeje pri každom kroku výpočtu. Krok výpočtu závisí na trojici $(p, a, X)$, kde:

- $p$ je aktuálny stav automatu
- $a$ je aktuálne čítaný vstupný symbol. Je možné uvažovať aj to, že sa aktuálny vstupný symbol ignoruje - v takom prípade $a = \varepsilon$.
- $X$ je aktuálny symbol na vrchu zásobníka. Tento symbol sa pri čítaní **vždy zo zásobníka vyberá**.

Na základe tejto trojice automat prejde do nejakého stavu $q$ a do zásobníka vloží reťazec symbolov $\alpha$. T.j. napríklad ak $(q_0, aZ_0) \in \delta(q_0, a, Z_0)$, tak potom je v ZA možný prechod zo stavu $q_0$ do stavu $q_0$ pri čítaní vstupného symbolu $a$, pričom na vrchu zásobníka musí byť $Z_0$, ktoré sa vyberie a nahradí reťazcom $aZ_0$.

Zásobníkový automat bude akceptovať **práve tie reťazce**, pre ktoré **existuje spôsob**, ako ich celé prečítať a skončiť v akceptačnom stave. Množina týchto reťazcov sa nazýva **jazyk akceptovaný zásobníkovým automatom** a formálne sa pre zásobníkový automat $P$ zapisuje:

$$L(P) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \alpha)\}$$

kde $q \in F$, t.j. $q$ je akceptačný stav, $\alpha \in \Gamma^*$, t.j. $\alpha$ je ľubovoľný obsah zásobníka v momente, keď je vstup $w$, zložený zo vstupných symbolov, celý prečítaný.

Uvažujme teda, ako ZA spracuje reťazec $ab$. Na začiatku je v stave $q_0$, v zásobníku je len symbol $Z_0$:

$$(q_0, ab, Z_0)$$

Podľa obrázka má teraz ZA dve možnosti:

- Zostať v stave $q_0$, pričom sa zo vstupu prečíta $a$, zo zásobníka sa vyberie $Z_0$ a nahradí sa $aZ_0$ t.j

$$(q_0, ab, Z_0) \vdash (q_0, b, aZ_0)$$

- Prípadne dôjde k prechodu do stavu $q_1$ bez čítania vstupu, pričom sa zo zásobníka vyberie $Z_0$ a vloží sa doň $Z_0$ (t.j. de facto sa obsah zásobníka nezmení)

$$(q_0, ab, Z_0) \vdash (q_1, ab, Z_0)$$

V prípade, že sa vykonal prechod:

$$(q_0, ab, Z_0) \vdash (q_1, ab, Z_0)$$

tak v ďalšom kroku je možný **jedine** prechod do stavu $q_2$, znovu sa počas neho ignoruje vstupný symbol, zo zásobníka sa vyberie $Z_0$ a vloží sa $Z_0$, t.j. celý výpočet:

$$(q_0, ab, Z_0) \vdash (q_1, ab, Z_0) \vdash (q_2, ab, Z_0)$$

Tu sa ZA zasekne v stave $q_2$. A hoci je $q_2$ akceptačný stav, vstup **nebol prečítaný** a teda ho v tomto výpočte **nepovažujeme za akceptovaný.**

V prípade, že sa prvý symbol vstupu prečítal:

$$(q_0, ab, Z_0) \vdash (q_0, b, aZ_0)$$

tak v ďalšom kroku je možný **jedine** prechod do stavu $q_1$, pričom sa počas neho ignoruje vstupný symbol, zo zásobníka sa vyberie $a$ a vloží sa $a$, t.j. výpočet:

$$(q_0, ab, Z_0) \vdash (q_0, b, aZ_0) \vdash (q_1, b, aZ_0)$$

V tomto momente je v ZA možné vykonať prechod zo stavu $q_1$ do stavu $q_1$, pričom na vstupe musí byť $b$ (je tam a prečíta sa) a na vrchu zásobníka musí byť $a$, pričom sa vyberie a nahradí sa $\varepsilon$(t.j. nenahradí sa ničím):

$$(q_0, ab, Z_0) \vdash (q_0, b, aZ_0) \vdash (q_1, b, aZ_0) \vdash (q_1, \varepsilon, Z_0)$$

Tak sme zistili, že po spracovaní reťazca *ab* sme sa ocitli v stave $q_1$:

$$(q_0, ab, Z_0) \vdash (q_0, b, aZ_0) \vdash (q_1, b, aZ_0) \vdash (q_1, \varepsilon, Z_0)$$

stav $q_1$ **nie je akceptačný.** Znamená to, že reťazec *ab* neakceptujeme?... áno, aj nie :). Tento výpočet - končiaci v stave $q_1$ **nie je akceptačný.** Avšak v automate ešte existuje možnosť ísť zo stavu $q_1$ do stavu $q_2$, pričom na vrchu zásobníka musí byť symbol $Z_0$ a vstup sa ignoruje. To teraz môžeme vykonať a dostávame výpočet:

$$(q_0, ab, Z_0) \vdash (q_0, b, aZ_0) \vdash (q_1, b, aZ_0) \vdash (q_1, \varepsilon, Z_0) \vdash (q_2, \varepsilon, Z_0)$$

Čiže vidíme, že v ZA je **možné** prečítať celý vstup *ab* a **dostať sa** do akceptačného stavu. Preto v tomto ZA vstup *ab* **akceptujeme.**

Akceptuje ZA reťazec $\varepsilon$ ? Zistíme to tak, že sa pokúsime nájsť výpočet, ktorý pre vstup $\varepsilon$ dokáže prejsť do stavu $q_2$, t.j. či platí, že :

$$(q_0, \varepsilon, Z_0) \vdash^* (q_2, \varepsilon, Z_0)$$

Naozaj, vďaka prechodom z $q_0$ do $q_1$, resp. z $q_1$ do $q_2$, ktoré ignorujú vstup a na vrchu zásobníka musí byť $Z_0$ je možné vykonať výpočet:

$$(q_0, \varepsilon, Z_0) \vdash (q_1, \varepsilon, Z_0) \vdash (q_2, \varepsilon, Z_0)$$

Preto tento ZA akceptuje aj reťazec $\varepsilon$.

Akceptuje ZA reťazec *ba* ? Na začiatku máme konfiguráciu $(q_0, ba, Z_0)$. Jediný krok výpočtu, ktorý môžeme urobiť, je:

$$(q_0, ba, Z_0) \vdash (q_1, ba, Z_0)$$

Jediný krok výpočtu, ktorý môžeme urobiť zo stavu $q_1$ je:

$$(q_0, ba, Z_0) \vdash (q_1, ba, Z_0) \vdash (q_2, ba, Z_0)$$

Čiže sa vieme pre reťazec *ba* dostať aj do akceptačného stavu, avšak sme neprečítali vstup *ba*. Dá sa ukázať, že pre tento reťazec **neexistuje spôsob**, ako ho prečítať a skončiť v akceptačnom stave $q_2$. Preto tento reťazec daný ZA **neakceptuje!**

Skúsme sa teda zamyslieť, aké reťazce daný zásobníkový automat akceptuje.

1. Budú to také reťazce, pre ktoré existuje akceptačný výpočet, t.j. celé sa prečítajú a automat skončí v akceptačnom stave $q_2$.
2. Na to, aby sme skončili v stave $q_2$, musíme pre daný vstup vedieť po jeho celom prečítaní skončiť aj v stave $q_1$, pričom na vrchu zásobníka musí byť $Z_0$.
3. V počiatočnom stave $q_0$ je možné čítať zo vstupu symboly $a$ - všimnite si, že po každom prečítanom symbole sa na vrch zásobníka vloží symbol $a$.
4. T.j. ak prečítam zo vstupu $n$-krát symbol $a$, tak na vrchu zásobníka bude $n$-krát symbol $a$.
5. Následne môžem "zadarmo" prejsť do stavu $q_1$, t.j. bez čítania vstupného symbolu vtedy, ak je na vrchu zásobníka $Z_0$ alebo $a$. Zároveň tento symbol na vrchu zásobníka ostane aj po prechode do stavu $q_1$.

(pokračovanie:)

6. V stave $q_1$ teraz môžem zo vstupu čítať symboly $b$, pričom pri každom čítaní musí byť zároveň na vrchu zásobníka symbol $a$, ktorý odstránim a nič do zásobníka nepridám.
7. Čiže ak sa v stave $q_1$ na vrchu zásobníka nachádza $n$-krát symbol $a$, tak môžem zo vstupu prečítať $n$-krát písmeno $b$ a $n$ symbolov $a$ zo zásobníka odstránim.
8. Aby sa teda stalo, že skončím v stave $q_1$ so zásobníkom, na vrchu ktorého je $Z_0$ a čítal som pri tom nejaké vstupné symboly, tak som musel najprv v $q_0$ prečítať $n$-krát $a$, potom sa presunúť do stavu $q_1$ a v ňom prečítať $n$-krát písmeno $b$.
9. Z toho je vidieť, že automat bude akceptovať určite také reťazce, ktoré sú v tvare $a^n b^n$, lebo

$$(q_0, a^n b^n, Z_0) \vdash^* (q_0, b^n, a^n Z_0) \vdash (q_1, b^n, a^n Z_0) \vdash^* (q_1, \varepsilon, Z_0) \vdash (q_2, \varepsilon, Z_0)$$

Navyše, ak by bol na vstupe iný reťazec, než v tvare $a^n b^n$, tak sa ZA určite:

- Alebo zasekne v stavoch $q_1, q_2$, pričom sa nedočíta celý vstup do konca, napr. pre vstupy $ba, bb, \ldots$
- Alebo sa síce dočíta celý vstup do konca, ale zasekne sa v stave $q_1$, napr pre vstup $aa, aab, \ldots$

Preto jazyk, ktorý akceptuje uvedený zásobníkový automat, je $L(P_1) = \{a^n b^n \mid n \geq 0\}$.

# Nájdite zásobníkové automaty

Nájdite zásobníkové automaty k jazykom:

1. $L_1 = \{w \in \{a,b\}^* \mid \sharp_a(w) = \sharp_b(w)\}$, t.j. reťazce z písmen $\{a,b\}$, v ktorých je počet symbolov $a, b$ rovnaký.
2. $L_2 = \{w \in \{a,b\}^* \mid \sharp_a(w) > \sharp_b(w)\}$, t.j. reťazce z písmen $\{a,b\}$, v ktorých je počet symbolov $a$ väčší než počet symbolov $b$.
3. $L_3 = \{w \in \{a,b\}^* \mid \sharp_a(w) < \sharp_b(w)\}$, t.j. reťazce z písmen $\{a,b\}$, v ktorých je počet symbolov $a$ menší než počet symbolov $b$.
4. $L_4 = \{wcw^R \mid w \in \{a,b\}^*\}$, t.j. palindrómy v strede ktorých je symbol $c$.
5. $L_5 = \{xaby \mid x \in \{a,b\}^*, y \in \{a,b\}^*\}$, t.j. reťazce z písmen $\{a,b\}$ obsahujúce $ab$ ako podreťazec
6. $L_6 = \{a^n b^n c^m \mid n \geq 0, m \geq 0\}$

# Príklad č. 1

Nájdite zásobníkový automat, ktorý akceptuje jazyk
$L_1 = \{w \in \{a, b\}^* \mid \sharp_a(w) = \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a, b$ rovnaký.

- Je zrejmé, že automat bude akceptovať len také reťazce, ktoré obsahujú rovnaký počet symbolov *a* a *b*.
- Keďže ZA číta vstup symbol po symbole, po prvom symbole automat vie, či prečítal *a* a teda niekde ďalej musí nasledovať *b*, alebo naopak.
- Preto zásobník vieme využiť ako pamäťové médium, pomocou ktorého si budeme pamätať, **koľko** "opačných" symbolov (k *a* je opačný *b* a naopak, k *b* je opačný *a*) ešte musíme prečítať, aby boli počty vyrovnané.

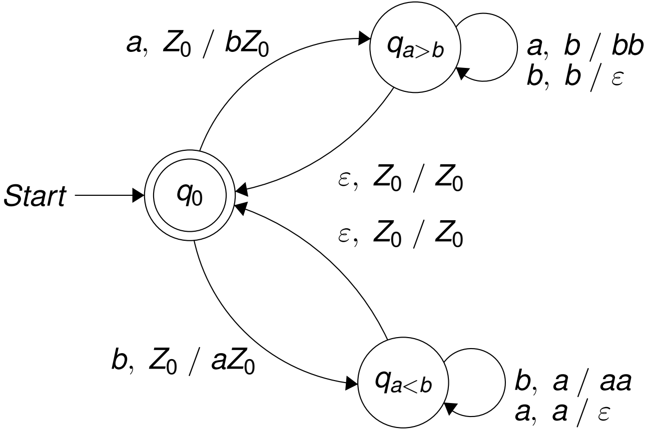
Pri spracúvaní reťazca by bolo vhodné rozlišovať 3 situácie vzhľadom na **doteraz prečítanú časť slova**:

- Počet doteraz prečítaných $a$ a $b$ bol rovnaký - taká je situácia na **začiatku** a taktiež je to situácia, v ktorej ak skončíme a vstup bol celý prečítaný, vstup **budeme akceptovať**. Teda ju bude reprezentovať **počiatočná a zároveň akceptačný** stav $q_0$.
- Počet doteraz prečítaných $a$ bol **väčší** než počet doteraz prečítaných $b$. To môže reprezentovať stav $q_{a>b}$. Zároveň to znamená, že aby sme slovo akceptovali, **musíme** prečítať ešte nejaké $b$-čka.
- Počet doteraz prečítaných $a$ bol **menší** než počet doteraz prečítaných $b$. To môže reprezentovať stav $q_{a<b}$. Zároveň to znamená, že aby sme slovo akceptovali, **musíme** prečítať ešte nejaké $a$-čka.

- Ak sme v stave $q_0$, na vrchu zásobníka je $Z_0$ a počet doteraz prečítaných symbolov je rovnaký. Ak teraz čítame na vstupe $a$, prejdeme do stavu $q_{a>b}$. Zároveň si do zásobníka vložíme symbol $b$, ktorý bude reprezentovať to, že musíme ešte prečítať jedno $b$-čko.
- Ak sme v stave $q_0$, na vrchu zásobníka je $Z_0$ a počet doteraz prečítaných symbolov je rovnaký. Ak teraz čítame na vstupe $b$, prejdeme do stavu $q_{a<b}$. Zároveň si do zásobníka vložíme symbol $a$, ktorý bude reprezentovať to, že musíme ešte prečítať jedno $a$-čko.
- Ak sme v stave $q_{a>b}$, znamená to, že na vrchu zásobníka je symbol $b$, t.j. čakáme *minimálne* na jedno $b$-čko, aby sa počty vyrovnali. Ak v tomto stave príde na vstupe $b$, tak toto $b$ zo zásobníka môžeme **odstrániť**. Naopak, ak v tomto stave zo vstupu čítame $a$, tak nám chýba ďalšie $b$ na vstupe, čo znázorníme tak, že znovu do zásobníka vložíme ďalšie $b$. Čiže v tomto stave platí, že koľkokrát je symbol $b$ v zásobníku, tak na toľko $b$ na vstupe čakáme, aby sa počty vyrovnali.

- Analogicky, ak sme v stave $q_{a<b}$, znamená to, že na vrchu zásobníka je symbol $a$, t.j. čakáme *minimálne* na jedno $a$-čko, aby sa počty vyrovnali. Ak v tomto stave príde na vstupe $a$, tak toto $a$ zo zásobníka môžeme **odstrániť**. Naopak, ak v tomto stave zo vstupu čítame $b$, tak nám chýba ďalšie $a$ na vstupe, čo znázorníme tak, že znovu do zásobníka vložíme ďalšie $a$. Čiže v tomto stave platí, že koľkokrát je symbol $a$ v zásobníku, tak na toľko $a$ na vstupe čakáme, aby sa počty vyrovnali.
- Ak sa v stavoch $q_{a>b}$ alebo $q_{a<b}$ stane, že sa na vrchu zásobníka ocitne symbol $Z_0$, znamená to, že sa nám doteraz prečítané vstupné symboly vyrovnali a môžeme sa vrátiť do stavu $q_0$.

Keď si uvedené pravidlá znázorníme obrázkom, dostaneme nasledovný obrázok zásobníkového automatu:



A formálne dostávame zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde:

- $Q = \{q_0, q_{a>b}, q_{a<b}\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{Z_0, a, b\}$
- $q_0$ je počiatočný stav
- $Z_0$ je počiatočný zásobníkový symbol
- $F = \{q_0\}$
- a prechodová funkcia $\delta$ je daná obrázkom, resp. aj predpisom (viď ďalší slajd)

Prechodová funkcia $\delta$:

- $\delta(q_0, a, Z_0) = \{(q_{a>b}, bZ_0)\}$
- $\delta(q_0, a, Z_0) = \{(q_{a<b}, aZ_0)\}$
- $\delta(q_{a>b}, a, b) = \{(q_{a>b}, bb)\}$
- $\delta(q_{a>b}, b, b) = \{(q_{a>b}, \varepsilon)\}$
- $\delta(q_{a>b}, \varepsilon, Z_0) = \{(q_0, Z_0)\}$
- $\delta(q_{a<b}, a, a) = \{(q_{a>b}, \varepsilon)\}$
- $\delta(q_{a<b}, b, a) = \{(q_{a>b}, aa)\}$
- $\delta(q_{a<b}, \varepsilon, Z_0) = \{(q_0, Z_0)\}$

Ak je zostrojený zásobníkový automat korektný, musia platiť 2 veci:

- Každý reťazec z jazyka $L_1$ (t.j. taký, ktorý obsahuje rovnaký počet $a$ a $b$ )musí byť v automate akceptovaný, t.j. $L_1 \subseteq L(P)$.
- Každý reťazec, ktorý automat akceptuje, musí patriť do jazyka $L_1$, t.j. $L(P) \subseteq L_1$.

Neformálny dôkaz toho, že každý reťazec, ktorý obsahuje rovnaký počet $a$ a $b$, má v automate akceptačný výpočet, t.j. že $L_1 \subseteq L(P)$:

- Každý reťazec $w$ , ktorý obsahuje rovnaký počet $a$ a $b$ sa dá rozdeliť na menšie segmenty $w_1, w_2, ..., w_n$, t.j. $w = w_1 w_2 ... w_n$, v ktorých tiež platí, že počet $a$ a $b$ je rovnaký.
- Toto delenie je robené tak, aby segment začínal napr. symbolom $a$ a končil symbolom $b$, ktorý je "párový" k prvému symbolu $a$, t.j. uzatvára segment, aby bol v ňom rovnaký počet symbolov $a$ a $b$. Analogicky naopak, ak segment začína symbolom $b$, tak končí párovým $a$-čkom.
- Napríklad v reťazci $w = abbbaa$ by bolo delenie $w = w_1 w_2$, kde $w_1 = ab$ a $w_2 = bbaa$.
- Automat je skonštruovaný tak, aby každý segment svojim prvým symbolom prešiel do stavu $q_{a>b}$ alebo $q_{a<b}$ a zo stavu sa vrátil do stavu $q_0$ po prečítaní svojho posledného symbolu (t.j. po vyrovnaní počtu symbolov).

Napríklad pre $w = abbbaa$, kde sme delenie určili ako $w = w_1 w_2$, $w_1 = ab$, $w_2 = bbaa$ by bol výpočet pre prvú časť $w_1 = ab$:

$$(q_0, abbbaa, Z_0) \vdash (q_{a>b}, bbbaa, bZ_0) \vdash (q_{a>b}, bbaa, Z_0) \vdash (q_0, bbaa, Z_0)$$

a následne pre druhú časť $w_2 = bbaa$:

$$(q_0, bbaa, Z_0) \vdash (q_{a<b}, baa, aZ_0) \vdash (q_{a<b}, aa, aaZ_0) \vdash (q_{a<b}, a, aZ_0) \vdash$$

$$\vdash (q_{a<b}, \varepsilon, Z_0) \vdash (q_0, \varepsilon, Z_0)$$

Teda spolu:

$$(q_0, abbbaa, Z_0) \vdash^* (q_0, \varepsilon, Z_0)$$

a teda reťazec *abbbaa* by bol akceptovaný.

Neformálny dôkaz toho, že každý akceptovaný reťazec obsahuje rovnaký počet $a$ a $b$, t.j. $L(P) \subseteq L_1$.

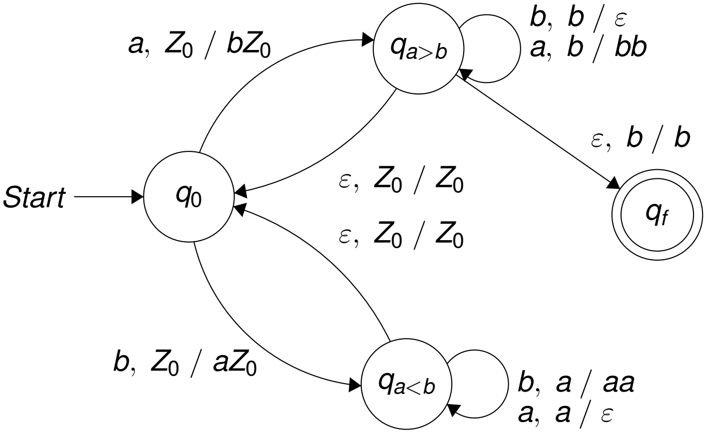
1. Akceptovaný reťazec je napríklad $\varepsilon$. Tento uvedenú vlastnosť spĺňa.
2. Ak akceptovaný reťazec obsahuje aspoň 1 symbol, tak po prečítaní prvého symbolu skončí v stave $q_{a>b}$ ak bol prvý symbol $a$, alebo v stave $q_{a<b}$, ak bol prvý symbol $b$.
3. Z tohto stavu **neodíde dovtedy**, kým sa nepodarí vyrovnať počty doteraz prečítaných $a$ a $b$. Pretože až ich vyrovnaním sa na vrch zásobníka dostane symbol $Z_0$, vďaka ktorému sa dá vrátiť do stavu $q_0$.
4. Teda každý akceptovaný reťazec sa dá rozdeliť na segmenty $w_1, w_2, ..., w_n$, pre ktoré platí, že ich čítaním sa automat dostane do stavu $q_{a>b}$ alebo $q_{a<b}$ a odchádza z neho až vtedy, keď je počet symbolov $a$ a $b$ rovnaký.
5. Teda segmenty obsahujú všetky rovnaký počet symbolov. A ich zreťazenie $w = w_1 w_2 ... w_n$ rovnako musí obsahovať **rovnaký počet symbolov**.

# Príklad č. 2

Nájdite zásobníkový automat, ktorý akceptuje jazyk $L_2 = \{w \in \{a, b\}^* \mid \sharp_a(w) > \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a$ väčší než počet symbolov $b$.

- Je zrejmé, že automat bude akceptovať len také reťazce, ktoré obsahujú väčší počet symbolov $a$ než $b$.
- Keby sme vychádzali z automatu, ktorý sme skonštruovali v predchádzajúcej úlohe, tak pre reťazce, ktoré majú väčší počet $a$ než $b$ platí, že po ich dočítaní viem skončiť v stave $q_{a>b}$ a zároveň v zásobníku **musí byť** aspoň jedno $b$-čko (lebo signalizuje, že ešte potrebujem $b$ prečítať, aby bol počet rovnaký).
- Preto malou úpravou automatu vieme dostať zásobníkový automat, ktorý bude akceptovať jazyk $L_2$.

- Vieme teda, že ak je na vstupe slovo, ktoré má viac $a$ než $b$, tak po jeho prečítaní výpočet skončí v stave $q_{a>b}$ a v zásobníku bude aspoň jeden symbol $b$.
- Niekomu by mohlo napadnút', že riešením je jednoducho **urobit'** z $q_{a>b}$ **akceptačný stav.**
- Problém s týmto riešením je, že v stave $q_{a>b}$ vieme skončit' aj pre slová, v ktorých je počet $a$ a $b$ rovnaký (pretože v pôvodnom automate sme potom prechodom, ktorý nečítal vstup, len potreboval, aby bol na vrchu zásobníka $Z_0$, prešli do akceptačného stavu $q_0$).
- Preto teraz potrebujeme pri akceptácii testovat', či je na vrchu zásobníka symbol $b$.
- Najjednoduchšie je pridat' do automatu nový stav, ktorý bude jediný **akceptačný stav.**
- Do tohto stavu prejdeme zo stavu $q_{a>b}$ bez čítania vstupného symbolu za podmienky, že na vrchu zásobníka je $b$.



Start
$q_0$
$a,\ Z_0\ /\ bZ_0$
$q_{a>b}$
$b,\ b\ /\ \varepsilon$
$a,\ b\ /\ bb$
$\varepsilon,\ b\ /\ b$
$q_f$
$\varepsilon,\ Z_0\ /\ Z_0$
$\varepsilon,\ Z_0\ /\ Z_0$
$b,\ Z_0\ /\ aZ_0$
$q_{a<b}$
$b,\ a\ /\ aa$
$a,\ a\ /\ \varepsilon$

A formálne dostávame zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde:

- $Q = \{q_0, q_{a>b}, q_{a<b}, q_f\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{Z_0, a, b\}$
- $q_0$ je počiatočný stav
- $Z_0$ je počiatočný zásobníkový symbol
- $F = \{q_f\}$
- a prechodová funkcia $\delta$ je daná obrázkom, resp. aj predpisom (viď ďalší slajd)

Prechodová funkcia $\delta$:

- $\delta(q_0, a, Z_0) = \{(q_{a>b}, bZ_0)\}$
- $\delta(q_0, a, Z_0) = \{(q_{a<b}, aZ_0)\}$
- $\delta(q_{a>b}, a, b) = \{(q_{a>b}, bb)\}$
- $\delta(q_{a>b}, b, b) = \{(q_{a>b}, \varepsilon)\}$
- $\delta(q_{a>b}, \varepsilon, Z_0) = \{(q_0, Z_0)\}$
- $\delta(q_{a<b}, a, a) = \{(q_{a>b}, \varepsilon)\}$
- $\delta(q_{a<b}, b, a) = \{(q_{a>b}, aa)\}$
- $\delta(q_{a<b}, \varepsilon, Z_0) = \{(q_0, Z_0)\}$
- $\delta(q_{a>b}, \varepsilon, b) = \{(q_f, b)\}$

Ak je zostrojený zásobníkový automat korektný, musia platiť 2 veci:

- Každý reťazec z jazyka $L_2$ (t.j. taký, ktorý obsahuje rovnaký počet $a$ a $b$ )musí byť v automate akceptovaný, t.j. $L_2 \subseteq L(P)$.
- Každý reťazec, ktorý automat akceptuje, musí patriť do jazyka $L_2$, t.j. $L(P) \subseteq L_2$.

Neformálny dôkaz toho, že každý reťazec, ktorý obsahuje väčší počet $a$ než $b$, má v automate akceptačný výpočet, t.j. že $L_2 \subseteq L(P)$:

- Každý reťazec $w$ , ktorý obsahuje väčší počet $a$ a $b$ sa dá rozdeliť na menšie segmenty $w_1, w_2, ..., w_n$, t.j. $w = w_1 w_2 ... w_n$, v ktorých tiež platí, že počet $a$ a $b$ je rovnaký v segmentoch $w_1, ..., w_{n-1}$ a v poslednom segmente je aspoň o 1 $a$ viac než $b$.
- Toto delenie je robené tak, aby segmenty $w_1$ až $w_{n-1}$ začínali napr. symbolom $a$ a končil symbolom $b$, ktorý je "párový" k prvému symbolu $a$, t.j. uzatvára segment, aby bol v ňom rovnaký počet symbolov $a$ a $b$. Analogicky naopak, ak segment začína symbolom $b$, tak končí párovým $a$-čkom.
- Napríklad v reťazci $w = abaabab$ by bolo delenie $w = w_1 w_2$, kde $w_1 = ab$ a $w_2 = aabab$.
- Automat je skonštruovaný tak, aby každý segment $w_1$ až $w_{n-1}$ svojim prvým symbolom prešiel do stavu $q_{a>b}$ alebo $q_{a<b}$ a zo stavu sa vrátil do stavu $q_0$ po prečítaní svojho posledného symbolu (t.j. po vyrovnaní počtu symbolov).
- A posledný segment sa dostane do $q_{a>b}$ v ktorom sa dočíta do konca, pričom v zásobníku bude po jeho dočítaní aspoň jedno $b$ na vrchu.

Napríklad pre $w = abaabab$, kde sme delenie určili ako $w = w_1 w_2$, $w_1 = ab$, $w_2 = aabab$ by bol výpočet pre prvú časť $w_1 = ab$:

$$(q_0, abbbaa, Z_0) \vdash (q_{a>b}, bbbaa, bZ_0) \vdash (q_{a>b}, bbaa, Z_0) \vdash (q_0, bbaa, Z_0)$$

a následne pre druhú časť $w_2 = aabab$:

$$(q_0, aabab, Z_0) \vdash (q_{a>b}, abab, bZ_0) \vdash (q_{a>b}, bab, bbZ_0) \vdash (q_{a>b}, ab, bZ_0) \vdash$$

$$\vdash (q_{a>b}, b, bbZ_0) \vdash (q_0, \varepsilon, bZ_0) \vdash (q_f, \varepsilon, bZ_0)$$

Teda spolu:

$$(q_0, abaabab, Z_0) \vdash^* (q_f, \varepsilon, Z_0)$$

a teda reťazec *abaabab* by bol akceptovaný.

Neformálny dôkaz toho, že každý akceptovaný reťazec obsahuje väčší počet $a$ než $b$, t.j. $L(P) \subseteq L_2$.

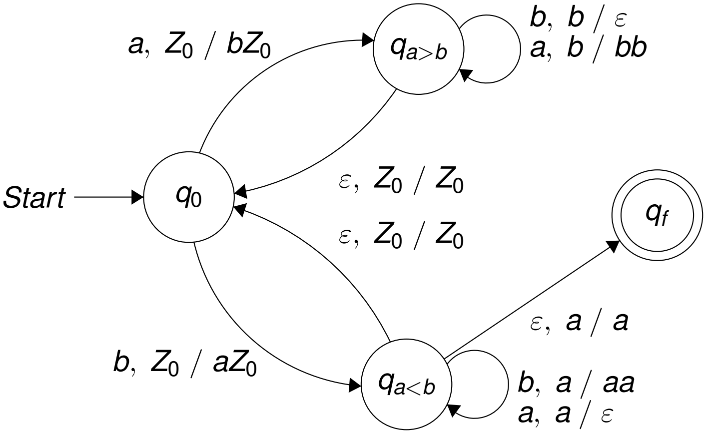
1. Akceptovaný reťazec má tú vlastnosť, že po jeho dočítaní musí automat skončiť v stave $q_{a>b}$ so symbolom $b$ na vrchu zásobníka, aby sa v ďalšom kroku vedel presunúť do $q_f$.
2. Aby sa dostal do stavu $q_{a>b}$, musí v stave $q_0$ prečítať aspoň jedno $a$ a následne, aby mu v stave $q_{a>b}$ zostal aspoň 1 symbol $b$ v zásobníku, tak nesmie prečítať toľko $b$, aby dorovnal prečítaný počet $a$.
3. Avšak do stavu $q_0$ sa už predtým mohol dostať tým ,že by prečítal segmenty vstupného slova $w_i$, v ktorých bol rovnaký počet symbolov $a$ a $b$.
4. Teda každý akceptovaný reťazec sa dá rozdeliť na segmenty $w_1, w_2, ..., w_n$, pre ktoré platí, že segmenty $w_1, ..., w_{n-1}$ obsahujú rovnaký počet $a$ a $b$ a ich čítaním sa vždy vie ZA vrátiť do $q_0$ a následne segment $w_n$ začína symbolom $a$, ktorým sa presunie do stavu $q_{a>b}$ a neobsahuje toľko symbolov $b$, aby sa počty dorovnali a v zásobníku zostalo $Z_0$ na vrchu.
5. Teda v zreťazení $w_1 w_2 ... w_{n-1} w_n$ bude aspoň o 1 $a$ **viac** než počet $b$.

# Príklad č. 3

Nájdite zásobníkový automat, ktorý akceptuje jazyk
$L_3 = \{w \in \{a, b\}^* \mid \sharp_a(w) < \sharp_b(w)\}$, t.j. reťazce z písmen $\{a, b\}$, v ktorých je počet symbolov $a$ menší než počet symbolov $b$.

- Tento zásobníkový automat bude **analógiou** automatu z predchádzajúcej úlohy, akurát že do akceptačného stavu nepôjde zo stavu $q_{a>b}$, ale zo stavu $q_{a<b}$ za situácie, že na vrchu zásobníka bude symbol $a$ signalizujúci, že bol prečítaný väčší počet symbolov $b$ než $a$.

Tu si vystačíme len s obrázkom:



# Príklad č. 4

Nájdite zásobníkový automat, ktorý akceptuje jazyk $L_4 = \{wcw^R \mid w \in \{a, b\}^*\}$, t.j. palindrómy v strede ktorých je symbol $c$.

Automat by mal akceptovať reťazce:

- $c$
- $aca$, $bcb$
- $aacaa$, $abcba$, $bacab$, $bbcbb$
- ...

Vidíme z popisu, že reťazce pozostávajú z 3 častí:

1. Reťazca $w$, ktorý tvoria všetky možné reťazce zo symbolov $\{a, b\}$
2. V strede je symbol $c$
3. reťazca $w^R$, ktorý predstavuje zrkadlový obraz (t.j. napísaný odzadu) reťazca $w$.

- Zásobníkový automat teda musí kontrolovať, či sa **za symbolom** $c$ nachádza zrkadlový obraz reťazca, ktorý je **pred symbolom** $c$.
- Keďže pred symbolom $c$ môže byť reťazec ľubovoľnej dĺžky, je potrebné si ho niekde zapamätať - ideálne v zásobníku.
- Navyše má zásobník výhodnú vlastnosť, že posledný vložený symbol je zároveň prvý vyberaný symbol, t.j. ak doň postupne ukladáme symboly reťazca $w$, tak ak zásobník čítame od vrchu, tak vlastne čítame priamo reťazec $w^R$.

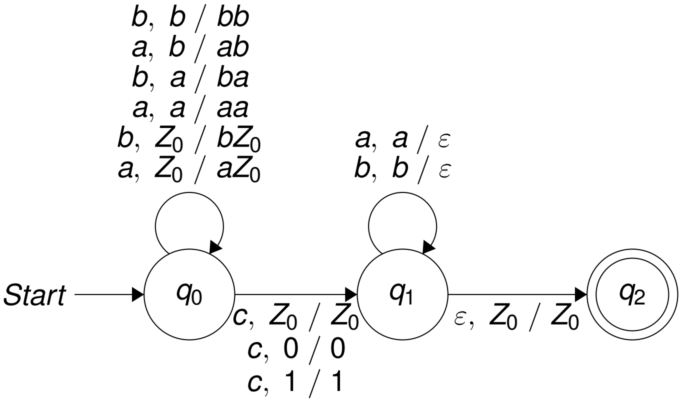
To znamená, že automat bude fungovať nasledovným spôsobom:

1. Najprv číta časť vstupu predstavujúcu reťazec $w$ pred symbolom $c$ - každý prečítaný symbol vloží do zásobníka. Toto bude predstavovať stav ZA $q_0$.
2. Následne, keď narazí na symbol $c$, tak sa prepne do režimu, v ktorom bude kontrolovať, či je na vstupe zrkadlový obraz reťazca $w$ tým, že bude porovnávať vstup s obsahom zásobníka - keďže na vstupe by **malo byť** $w^R$ a v zásobníku sa momentálne nachádza odvrchu **práve** $w^R$. Túto kontrolu, či sa obsah zásobníka zhoduje so vstupom, bude predstavovať stav $q_1$
3. Ak sa podarí *napárovať* každý vstupný symbol so zásobníkom, tak sa v momente, keď sa celý vstup prečíta v zásobníku na vrchu nachádza symbol $Z_0$, čo signalizuje akceptáciu slova. Tú bude signalizovať akceptačný stav $q_2$.

Pre zhrnutie

- ZA bude mať stav $q_0$ v ktorom bude zo vstupu čítať symboly $a, b$ z časti $w$ vstupu a ukladať ich do zásobníka.
- V momente, keď prečíta zo vstupu $c$, bude vedieť, že ďalej by mala nasledovať časť $w^R$ vstupu a prepne sa do stavu $q_1$. Pri tomto prepínaní sa do zásobníka nič nové nevkladá.
- V stave $q_1$ číta vstupné symboly a porovnáva ich s obsahom zásobníka. Ak sa zhodujú, odstráni symbol zo zásobníka a pokračuje s ďalším vstupným symbolom.
- Keď v stave $q_1$ dočíta celý vstup a skončí so zásobníkom na vrchu ktorého je len symbol $Z_0$, tak to signalizuje, že vstup je v požadovanom tvare. V takom prípade sa prepne do akceptačného stavu $q_2$.

Grafická reprezentácia takého ZA by bola:



A formálne dostávame zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde:

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b, c\}$
- $\Gamma = \{Z_0, a, b\}$
- $q_0$ je počiatočný stav
- $Z_0$ je počiatočný zásobníkový symbol
- $F = \{q_2\}$
- a prechodová funkcia $\delta$ je daná obrázkom, resp. aj predpisom (viď ďalší slajd)

Prechodová funkcia $\delta$:

- $\delta(q_0, a, Z_0) = \{(q_0, aZ_0)\}$
- $\delta(q_0, b, Z_0) = \{(q_0, bZ_0)\}$
- $\delta(q_0, a, a) = \{(q_0, aa)\}$
- $\delta(q_0, a, b) = \{(q_0, ab)\}$
- $\delta(q_0, b, a) = \{(q_0, ba)\}$
- $\delta(q_0, b, b) = \{(q_0, bb)\}$
- $\delta(q_0, c, Z_0) = \{(q_1, Z_0)\}$
- $\delta(q_0, c, a) = \{(q_1, a)\}$
- $\delta(q_0, c, b) = \{(q_1, b)\}$
- $\delta(q_1, a, a) = \{(q_1, \varepsilon)\}$
- $\delta(q_1, b, b) = \{(q_1, \varepsilon)\}$
- $\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$

Ak je zostrojený zásobníkový automat korektný, musia platiť 2 veci:

- Každý reťazec z jazyka $L_4$ (t.j. palindrómy v strede ktorých je $c$ )musí byť v automate akceptovaný, t.j. $L_4 \subseteq L(P)$.
- Každý reťazec, ktorý automat akceptuje, musí patriť do jazyka $L_4$, t.j. $L(P) \subseteq L_4$.

Neformálny dôkaz toho, že každý palindróm, ktorý obsahuje v strede $c$, má v automate akceptačný výpočet, t.j. že $L_4 \subseteq L(P)$:

- Každý palindróm, ktorý má v strede $c$, je v tvare $wcw^R$.
- V automate je možnosť pre časť reťazca $w$ túto časť v stave $q_0$ čítať a vkladať do zásobníka. Tým sa v zásobníku postupne od vrchu po spodok vytvorí reťazec $w^R$.
- Následne sa prečítaním symbolu $c$ automat prepne do stavu $q_1$.
- V tomto stave sa čítaním $w^R$ zo vstupu a porovnávaním $w^R$ v zásobníku postupne číta vstup a vyprázdňuje zásobník.
- Následne po prečítaní posledného symbolu a vyprázdnení posledného $\{a, b\}$zo zásobníka na vrchu zásobníka ocitne len $Z_0$, čím je umožnený prechod do akceptačného stavu $q_2$
- T.j. ak je na vstupe $wcw^R$, teda slovo z jazyka $L_4$, tak v automate existuje akceptačný výpočet:

$$(q_0, (wcw^R), Z_0) \vdash^* (q_0, cw^R, w^R Z_0) \vdash (q_1, w^R, w^R Z_0) \vdash^* (q_1, \varepsilon, Z_0) \vdash (q_2, \varepsilon, Z_0)$$

Neformálny dôkaz toho, že každý akceptovaný reťazec je zároveň palindrómom $wcw^R$.

1. Predpokladajme, že automat akceptuje nejaký reťazec $x$. To znamená, že $(q_0, x, Z_0) \vdash^* (q_2, \varepsilon, Z_0)$
2. To zároveň znamená, že aj $(q_0, x, Z_0) \vdash (q_1, \varepsilon, Z_0)$, teda že prečítaním tohto reťazca sa vie dostať do $q_1$.
3. Zamyslime sa, čo musel byť posledný čítaný symbol: ak ním bolo $c$, tak to znamená, že bol vykonaný prechod do stavu $q_1$. Keďže v zásobníku je po tomto prechode $Z_0$, muselo tam byť aj **predtým**.
4. A teda pred symbolom $c$ **nebolo nič iné čítané zo vstupu** - v opačnom prípade by to bolo totiž v zásobníku.
5. Teda vstup musel byť len reťazec $c$ - a ten je palindrómom.
6. Čo ak v kroku 3 nebol posledný čítaný symbol $c$ - muselo to byť $a$ alebo $b$.

(pokračovanie)

7. V takom prípade však musel byť prechod z predchádzajúceho stavu $q_1$ a teda prechod, ktorý je možný, len ak je v zásobníku rovnaký symbol ako na vstupe - t.j. prechod, ktorý odstraňuje zo zásobníka vrchný symbol.
8. Túto úvahu vieme opakovať, kým nedospejem do situácie, že sme do stavu $q_1$ prešli čítaním symbolu $c$ zo stavu $q_0$. Čiže prechody zo stavu $q_1$ do stavu $q_1$ boli umožnené len ak bol na vstupe taký reťazec, ktorý bol zároveň v zásobníku - nech je tento reťazec $w$.
9. Pred týmto reťazcom sa na vstupe musel teda nachádzať symbol $c$, ktorý spôsobil prechod do stavu $q_1$ zo stavu $q_0$.
10. Aby bol reťazec akceptovaný, tak počas jeho čítania v stave $q_0$ sa každý symbol vkladal do zásobníka, t.j. ak je v zásobníku pri prechode do $q_1$ reťazec $w$, tak sa tam musel dostať čítaním v stave $q_0$ - z dôvodu toho, ako funguje zásobník sa musel čítať "odzadu", t.j. ako $w^R$.
11. To znamená, že ak je nejaký reťazec akceptovaný, tak musí byť tvaru $w^R cw = wcw^R$, t.j. **práve** palindróm so stredným symbolom $c$.

# Príklad č. 5

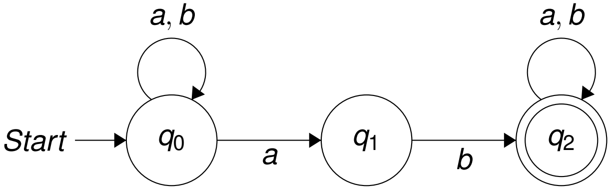
Nájdite zásobníkový automat, ktorý akceptuje jazyk
$L_5 = \{xaby \mid x \in \{a, b\}^*, y \in \{a, b\}^*\}$, t.j. reťazce z písmen $\{a, b\}$ obsahujúce $ab$ ako podreťazec.
Automat by mal akceptovať reťazce pozostávajúce z 3 častí:

- Ľubovoľného prefixu zo symbolov $a, b$
- Podreťazca $ab$
- Ľubovoľného sufixu zo symbolov $a, b$

Trikom pri konštrukcii tohto zásobníkového automatu je uvedomenie si nasledovnej skutočnosti:

**Daný jazyk je regulárny! To znamená, že stačí zostrojiť DKA / NKA / $\varepsilon$-NKA, ktorý daný jazyk akceptuje a pridať k nemu prácu so zásobníkom, pri ktorej sa zásobník úplne ignoruje**

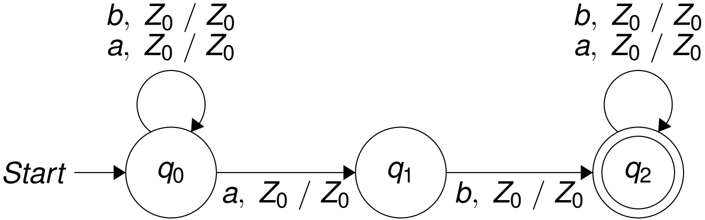
NKA akceptujúci uvedený jazyk je napríklad:



Keďže my chceme zásobníkový automat, ku každému prechodu pridáme prácu so zásobníkom, ktorá vezme z vrchu zásobníka symbol $Z_0$ a vzápätí ho vráti naspäť.

Tým v podstate modelujeme situáciu, že sa zásobník ignoruje.

Grafická reprezentácia výsledného ZA by bola:



A formálne dostávame zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde:

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{Z_0\}$
- $q_0$ je počiatočný stav
- $Z_0$ je počiatočný zásobníkový symbol
- $F = \{q_2\}$
- a prechodová funkcia $\delta$ je daná obrázkom, resp. aj predpisom (viď ďalší slajd)

Prechodová funkcia $\delta$:

- $\delta(q_0, a, Z_0) = \{(q_0, Z_0), (q_1, Z_0)\}$
- $\delta(q_0, b, Z_0) = \{(q_0, Z_0)\}$
- $\delta(q_1, b, Z_0) = \{(q_2, Z_0)\}$
- $\delta(q_2, a, Z_0) = \{(q_2, Z_0)\}$
- $\delta(q_2, b, Z_0) = \{(q_2, Z_0)\}$

Ak je zostrojený zásobníkový automat korektný, musia platiť 2 veci:

- Každý reťazec z jazyka $L_5$ (t.j. reťazce v tvare $xaby$, kde $x, y$ sú reťazce symbolov $\{a, b\}$) musí byť v automate akceptovaný, t.j. $L_5 \subseteq L(P)$.
- Každý reťazec, ktorý automat akceptuje, musí patriť do jazyka $L_5$, t.j. $L(P) \subseteq L_5$.

Neformálny dôkaz toho, že každý reťazec v tvare *xaby* má v automate akceptačný výpočet, t.j. že $L_5 \subseteq L(P)$:

- Pre prvú časť vstupného reťazca označenú $x$ platí, že pri jej čítaní ostáva automat v stave $q_0$.
- Následne sa podreťazcom *ab* postupne prepne zo stavu $q_0$ do stavu $q_1$ prečítaním $a$, a prečítaním $b$ sa prepne z $q_1$ do $q_2$.
- Pre zvyšnú časť vstupu označenú $y$ platí, že sa dá prečítať a stále zotrvať v akceptačnom stave $q_2$.
- Preto platí, že sa dá taký reťazec celý prečítať a skončiť v akceptačnom stave $q_2$, t.j. existuje výpočet:

$$(q_0, xaby, Z_0) \vdash^* (q_0, aby, Z_0) \vdash (q_1, by, Z_0) \vdash (q_2, y, Z_0) \vdash^* (q_2, \varepsilon, Z_0)$$

Neformálny dôkaz toho, že každý akceptovaný reťazec je zároveň reťazec v tvare $xaby$, kde $x, y$ sú ľubovoľné reťazce zo symbolov $\{a, b\}$.

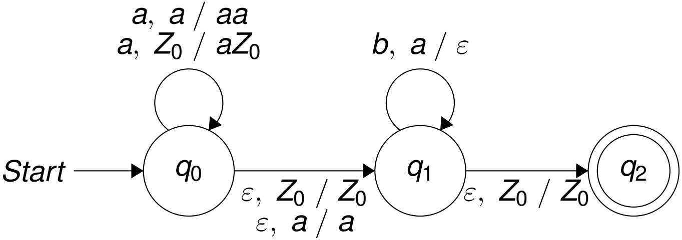
1. Predpokladajme, že automat akceptuje nejaký reťazec $w$. To znamená, že $(q_0, w, Z_0) \vdash^* (q_2, \varepsilon, Z_0)$
2. Z pohľadu na automat vidíme, že každý akceptovaný reťazec musí mať tú vlastnosť, že počas jeho spracovania sa prejde zo stavu $q_0$ cez $q_1$ do stavu $q_2$.
3. To zároveň znamená, že reťazec **musí** niekde obsahovať postupnosť *ab*.
4. Navyše vidíme, že ak sa už dostane do stavu $q_2$, tak potom čokoľvek, čo obsahuje, je možné bez problémov prečítať, t.j. za reťazcom *ab* môže nasledovať čokoľvek.
5. Taktiež vidíme, že ak pred *ab* obsahoval čokoľvek, tak sme to v podstate mohli prečítať a stále zotrvávať v stave $q_0$.
6. Preto každý akceptovaný reťazec spĺňa tvar $xaby$, kde $x, y$ sú ľubovoľné (aj prázdne) reťazce zo symbolov $\{a, b\}$.

# Príklad č. 6

Nájdite zásobníkový automat, ktorý akceptuje jazyk $L_6 = \{a^n b^n c^m \mid n \geq 0, m \geq 0\}$. Automat by mal akceptovať reťazce pozostávajúce z 2 hlavných častí:

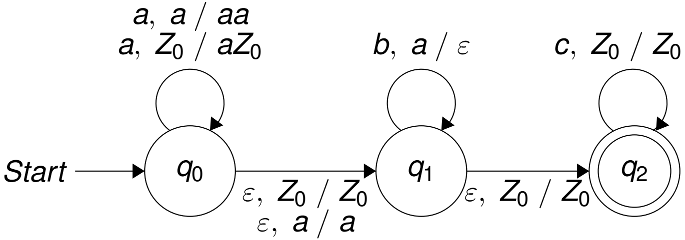
- Reťazca zo symbolov $a, b$, ktorých je rovnako veľa a navyše sú v tvare, že sú najprv symboly $a$ a potom symboly $b$.
- Reťazca zo symbolov $c$, ktorých je nejaký počet.
- Navyše platí, že počet symbolov $c$ nijako nezávisí na počte $a$, resp. $b$.

Keďže reťazce, ktoré má automat akceptovať, by mali začínať časťou, ktorá je v tvare $a^n b^n$, môžeme si pri konštrukcii ZA pomôcť zásobníkovým automatom pre tento jazyk (riešili sme ho v úlohe na slajde č. 61). V tomto automate platí, že ak má na vstupe reťazec $a^n b^n$, tak existuje výpočet, ktorý skončí v stave $q_2$ a na vrchu zásobníka je symbol $Z_0$:



- V uvedenom automate teda platí, že do akceptačného stavu $q_2$ sa vieme dostať a prečítať pri tom reťazec $a^n b^n$.
- Keďže my chceme akceptovať reťazce tvaru $a^n b^n c^m$, tak vlastne potrebujeme zo stavu $q_2$ ešte prečítať zvyšok reťazca, t.j. $c^m$.
- Keďže týchto $c$-čok je ľubovoľný počet a navyše je tento počet nezávislý od iných hodnôt, v podstate ani zásobník nepotrebujeme, t.j. môžeme ignorovať jeho obsah - keďže v stave $q_2$ je na jeho vrchu $Z_0$, tak ho tam necháme a len v slučke dočítame symboly $c$ do konca.
- Preto stačí len pridať do stavu $q_2$ slučku, v ktorej budeme čítať symboly $c$, pričom z vrchu zásobníka vezmeme $Z_0$ a zase ho naspäť vložíme.

Grafická reprezentácia výsledného ZA by bola:



A formálne dostávame zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde:

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b, c\}$
- $\Gamma = \{Z_0, a\}$
- $q_0$ je počiatočný stav
- $Z_0$ je počiatočný zásobníkový symbol
- $F = \{q_2\}$
- a prechodová funkcia $\delta$ je daná obrázkom, resp. aj predpisom (viď ďalší slajd)

Prechodová funkcia $\delta$:

- $\delta(q_0, a, Z_0) = \{(q_0, aZ_0)\}$
- $\delta(q_0, a, a) = \{(q_0, aa)\}$
- $\delta(q_0, \varepsilon, Z_0) = \{(q_1, aZ_0)\}$
- $\delta(q_0, \varepsilon, a) = \{(q_1, a)\}$
- $\delta(q_1, b, a) = \{(q_1, \varepsilon)\}$
- $\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$
- $\delta(q_2, c, Z_0) = \{(q_2, Z_0)\}$

Ak je zostrojený zásobníkový automat korektný, musia platiť 2 veci:

- Každý reťazec z jazyka $L_6$ (t.j. reťazce v tvare $a^n b^n c^m$) musí byť v automate akceptovaný, t.j. $L_6 \subseteq L(P)$.
- Každý reťazec, ktorý automat akceptuje, musí patriť do jazyka $L_6$, t.j. $L(P) \subseteq L_6$.

Neformálny dôkaz toho, že každý reťazec v tvare $a^n b^n c^m$ má v automate akceptačný výpočet, t.j. že $L_6 \subseteq L(P)$:

- Pre prvú časť vstupného reťazca, t.j. $a^n$ existuje výpočet, ktorý každé prečítané $a$ v stave $q_0$ vloží do zásobníka.
- Následne sa automat prepne do stavu $q_1$, v ktorom sa každé prečítané $b$ porovná so symbolom v zásobníku a ak je na vstupe $b$ a v zásobníku $a$, tak sa $a$ odstráni zo zásobníka a na vstupe sa prejde na ďalšie $b$. Ak je ich rovnaký počet, t.j na vstupe je $b^n$, tak na vrchu zásobníka sa ocitne $Z_0$.
- Následne je možné prejsť do stavu $q_2$ vďaka symbolu $Z_0$ na vrchu zásobníka. V tomto stave je možné prečítať ľubovoľnú postupnosť $c$-čok na vstupe, t.j. $c^m$.
- Dokopy teda ak je na vstupe $a^n b^n c^m$, tak existuje spôsob, ako ho prečítať a skončiť v akceptačnom stave.

$$(q_0, a^n b^n c^m, Z_0) \vdash^* (q_0, b^n c^m, a^n Z_0) \vdash (q_1, b^n c^m, a^n Z_0) \vdash^* (q_1, c^m, Z_0) \vdash$$

$$\vdash (q_2, c^m, Z_0) \vdash^* (q_2, \varepsilon, Z_0)$$

Neformálny dôkaz toho, že každý akceptovaný reťazec je zároveň reťazec v tvare $a^n b^n c^m$:

1. Predpokladajme, že automat akceptuje nejaký reťazec $w$. To znamená, že $(q_0, w, Z_0) \vdash^* (q_2, \varepsilon, Z_0)$
2. Z pohľadu na automat vidíme, že do akceptačného stavu $q_2$ prejdeme zo stavu $q_1$ len ak je na vrchu zásobníka $Z_0$.
3. To znamená, že by sme z $q_0$ prešli do $q_1$ a následne do $q_2$ bez čítania vstupu, t.j. prečítame $\varepsilon$
4. Ak by v zásobníku boli v stave $q_1$ iné symboly, než $Z_0$, tak tam musí byť symbol $a$. Ten je možné odstrániť len čítaním $b$ zo vstupu v stave $q_1$. Ale $a$ sa do zásobníka dostane len tak, že v stave $q_0$ čítame zo vstupu $a$. Aby sme prečítaním $b$-čok v $q_1$ odstránili všetky $a$ zo zásobníka, tak sme museli prečítať práve $a^n b^n$.
5. Následne je po prečítaní $a^n b^n$ možné prejsť do $q_2$. Každý reťazec, ktorý sa dočíta do konca môže teoreticky na konci obsahovať nejaké $c$ symboly, t.j. $c^m$, ktoré je možné zadarmo prečítať.
6. Preto každý akceptovaný reťazec zároveň spĺňa tvar $a^n b^n c^m$.

# Použitá literatúra

1 Hopcroft, Motwani, Ullman - Introduction to Automata Theory, Languages and Computations, 3rd Ed.