ZADÁNÍ SEMESTRÁLNÍ PRÁCE
GENEROVÁNÍ KONEČNÉHO AUTOMATU

Zadání

Naprogramujte v ANSI C přenositelnou[1] **konzolovou aplikaci**, která načte ze souboru definici konečného automatu, který přijme textový řetězec popsaný regulárním výrazem. A vygeneruje validní zdrojový kód v ANSI C, který bude představovat implementaci daného konečného automatu.

Program se bude spouštět příkazem: `fsmgen.exe` ⟨`file`⟩. Symbol ⟨`file`⟩ zastupuje jméno souboru s popisem konečného stavového automatu (popis vstupního souboru bude uveden dále). Váš program tedy může být během testování spuštěn například takto:

```
...\>fsmgen.exe graph.gv
```

Výstupem programu bude validní zdrojový kód v ANSI C, představující implementaci daného automatu. Pokud nebude 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 jsou pouze argumenty 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. Bude-li některá z částí chybět, kontrolní skript Vaši práci odmítne.

Specifikace výstupu programu

Výstupem programu budou soubor `fsm.c` jehož přeložením vznikne spustilený soubor `fsm.exe`. Výsledný program poté bude spuštěn příkazem `fsm.exe` ⟨`string`⟩. Symbol ⟨`string`⟩ představuje alfanumerický řetězec, který musí konečný automat zpracovat. U vstupního řetězce záleží na velikosti písmen. V případě, že byl řetězec zpracován úspěšně bude návratová hodnota programu 0, pokud se řetězec nepodařilo zpracovat, bude návratová hodnota programu 1. Při testování budou ověřovány obě možnosti.

Specifikace vstupu programu

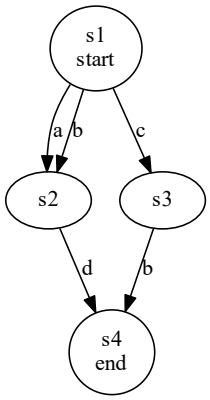
Vstupem programu jsou pouze parametry na příkazové řádce. Předaný parametr představuje jméno vstupního souboru. Jedná se o textový soubor.

[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 10 Buster s jádrem verze 4.19.0-11-amd64 a s překladačem gcc 8.3.0.

Popis vstupního souboru

Konečný automat bude popsán pomocí jazyka DOT pro nástroj Graphviz[2]. Vstupní soubor může vypadat například takto:

```
digraph G
{
⟶//nodes
⟶s1[label=start];
⟶s2;
⟶s3;
⟶s4[label=end];
⟶//edges
⟶s1->s2[label=a];
⟶s1->s3[label=c];
⟶s1->s2[label=b];
⟶s3->s4[label=b];
⟶s2->s4[label=d];
}
```



První řádek je vždy deklarace grafu. Na druhém řádku je vždy komentář, který označuje začátek seznamu uzlů grafu, každý uzel představuje jeden stav konečného automatu. Počáteční stav konečného automatu je označen řetězcem `[label=start];` a koncový stav je označen řetězcem `[label=end];`. Symbol ⟶ představuje odsazení textu, jedná se o řetězec bílých znaků (mezera, tabulátor).

Automat obsahuje pouze jediný počáteční stav a může obsahovat více koncových stavů. Délka jména uzlu nepřesáhne 10 znaků. Každý řádek bude zakončen sekvencí `\r\n`. Po seznamu uzlů následuje komentář, který označuje začátek seznamu hran grafu. Každá hrana pak představuje přechod mezi stavy konečného automatu.

Hrana je zapsána ve formátu `(startovní uzel)->(koncový uzel)[label=(symbol)]`. Kde `(startovní uzel)` a `(koncový uzel)` představují jeden z uzlů definovaných v seznamu uzlů a `(symbol)` je jeden alfanumerický znak, který aktivuje přechod po hraně. U znaku označujícího hranu záleží na velikosti. Startovní a koncový uzel hrany může být totožný uzel `(s1->s1[label=o])`. Mezi dvěma uzly může vést více nezávislých hran. Za poslední hranou následuje uzavírací závorka.

Užitečné techniky a odkazy

Uvedené techniky je možné (ale nikoliv nezbytně nutné) využít při řešení úlohy. Protože se jedná o postupy víceméně standardní, lze k nim nalézt velké množství dokumentace:

1. zpracování regulárních výrazů,
2. konečné automaty,
3. generování kódu.

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

[2] http://www.graphviz.org/doc/info/lang.html