Dokumentace úlohy SYN: Zvýraznění syntaxe v Perlu do IPP 2011/2012
Jméno a příjmení: Vojtěch Dlápal
Login: xdlapa01

Úvod

Úlohou je provést zvýraznění syntaxe ve vstupním souboru pomocí předem definovaných značek, a to podle regulárních výrazů spárovaných s množinou značek ve vstupním souboru s formátováním. Jako implementační jazyk byl zadán Perl, a proto se implementace snaží co nejlépe využít jeho předností, mezi které patří především práce s regulárními výrazy a s tím spojené funkce. Naopak zase obchází zdánlivé nedostatky, což bude zřejmé z dalšího popisu.

Dekompozice proběhla pomocí delegování partikulárních úloh na samostatné procedury. Jejich počet čítá šest a postupně slouží pro vytvoření řetězců se zahajovacími a ukončovacími značkami, výpis nápovědy, zpracování chyby při načítání parametrů, zpracování parametrů, zpracování vstupního souboru s formátováním a načtení vstupního souboru do řetězce. Jádro úlohy, čili vyhledání regulárních výrazů ve vstupním textu a vložení náležitých značek na správné pozice, provádí hlavní tělo skriptu.

Zpracování vstupních parametrů

Jako první úloha je provedeno zpracování parametrů. To se děje v proceduře `getParams,` a to pomocí funkce `GetOptions` z knihovny `Getopt::Long`, která byla nakonfigurována pro neshlukování zkrácených parametrů a propouštění parametrů chybně zadaných (pro pozdější zpracování). Nad rámec možností použité knihovny je ošetřeno vícenásobné zadání parametru a zadání parametru nesprávného. Pro tyto případy je volána procedura `badoptHandle`, která vypíše chybové hlášení a ukončí skript s příslušným návratovým kódem. Dále je navíc ošetřena unikátnost parametru `--help`.

Zpracování formátovacího souboru

Toto je jedna z nejkomplikovanějších částí skriptu a provádí ji procedura `parseFormat`. Cílem je naplnit globální datovou strukturu `format_hash` dvojicemi upravený regulární výraz – řetězec pro vytvoření značek. Nejdříve je třeba otevřít vstupní soubor s formátováním. Pokud tento nebyl zadán, procedura se ukončí a `format_hash` zůstane prázdný. Pokud se soubor nepodaří otevřít, anebo nesplňuje požadavky na vnitřní strukturu, je vypsáno chybové hlášení a skript je ukončen s daným návratovým kódem.

Pomocí funkce `split` je oddělen regulární výraz od řetězce pro formátování. Zadaný regulární výraz je poté pevně danou posloupností substitucí (nahrazování za pomocí regulárních výrazů) transformován na výraz odpovídající syntaxi Perlu pro pozdější použití vyhledávání vzorů. Následně je výsledný výraz zkontrolován pomocí jeho provedení funkcí `eval` a kontrolováním přítomnosti nepovolených vzorů (při chybě je skript ukončen s chybovým hlášením a daným návratovým kódem).

Speciálním případem je znak %, který má v kontextu vstupního regulárního výrazu speciální význam a znak “%” je značen jako %%. Proto jsou výskyty %% nahrazeny za speciální sekvenci `&&037prc&&,` a tedy je možné dále znak % uvažovat pouze v jeho speciálním významu. Sekvence je na znak % opět převedena až jako poslední stupeň nahrazovací kaskády.

Nakonec je jako hodnota klíče v `format_hash` uložen řetězec s formátováním, a to konkatenací za stávající řetězec. Pokud klíč ještě neexistuje, bude vytvořen. Datová struktura hash bohužel nezachovává pořadí záznamů podle okamžiku vložení a zadání požaduje dodržení pořadí aplikace jednotlivých regulárních výrazů tak, jak byly zapsány v souboru pro formátování. Proto je výsledný regulární výraz uložen také do pole, kde index značí jeho pořadí.

Vkládání značek do výstupního souboru

V hlavním těle skriptu je po zavolání procedur pro zpracování parametrů a formátovacího souboru dále zavolána procedura, která načte celý vstup do řetězce `inText`. Poté je pro každou pozici vstupního textu inicializována položka pole. Dále je iterováno přes pole pořadí regulárních výrazů, pro každý výraz je ve `format_hash` nalezen řetězec formátovacích příkazů, ten je předán proceduře `parseTages`, která naplní řetězce `endTags` a `startTags` náležitou posloupností značek. Poté jsou ve vstupním textu vyhledány vzory podle zadaných regulárních výrazů a pozice shod jsou u všech neprázdných délek uloženy do polí `endPos` a `startPos`. Pro každou pozici jsou do pole `tagsOnIndexes` přidány nově získané značky.

Nakonec je otevřen výstupní soubor (s ošetřením selhání) a vstupní soubor je zapisován po znaku, což je implementováno pomocí počítaného cyklu a funkce `substr`, která pro každou další iteraci vybere další podřetězec řetězce vstupního textu `inText` o délce jedna`.` Pro každý znak je zkontrolováno, zda pro daný index nemají být vloženy značky z `tagsOnIndexes`. Pokud ano, tak se tyto vytisknou do výstupního souboru ještě před znakem.

.