Dokumentace úlohy SYN: Zvýraznění syntaxe v PHP 5 do IPP 2015/2016
Jméno a příjmení: Nikola Valešová
Login: xvales02

Analýza zadání

Mým úkolem bylo napsat skript, který provede zvýraznění syntaxe ve vstupním souboru a výsledek uloží do výstupního souboru, přičemž zvýrazněním syntaxe je míněno přidání formátovacích příkazů před a za takovou část vstupního textu, která odpovídá regulárnímu výrazu popsanému ve formátovacím souboru.

Celý projekt je možno rozdělit do několika logických celků: zpracování argumentů, ověření validnosti formátovacího souboru, úprava zadaných regulárních výrazů na ekvivalentní regulární výrazy v jazyce PHP, vyhledání odpovídajících řetězců ve vstupním textu a přidání formátovacích příkazů.

Zpracování argumentů

Zadané argumenty se prochází pomocí funkce `foreach` a vždy se kontroluje, zda se jedná o validně zadaný argument. Pokud je zadaný nevhodný počet nebo formát argumentů, volá se funkce `print_error`, která vypíše chybové hlášení na standardní chybový výstup, a skript skončí s příslušnou návratovou hodnotou. Pokud jsou argumenty zadány správně, pomocí funkcí `strstr` a `substr` je získána cesta k souboru a jeho název, což se uloží do asociativního pole, kde je klíčem název parametru.

Ověření validnosti formátovacího souboru

V první části se prochází všechny formátovací příkazy načtené z formátovacího souboru a kontroluje se, zda se jedná o platný příkaz a také jestli se na každém řádku vyskytuje daný příkaz nejvýše jednou. Pokud je nějaká z těchto podmínek porušena, skript končí s návratovou hodnotou čtyři. V opačném případě přechází do druhé části kontroly formátovacího souboru, kde se ověřuje platnost zadaných regulárních výrazů. Ověřuje se počet závorek a kombinace sousedních znaků.

Úprava regulárních výrazů

Regulární výrazy je potřeba převést na ekvivalentní, přijímané funkcemi jazyka PHP. Většina operátorů má v PHP odpovídající náhradu, je pouze třeba ošetřit, zda není význam znaku zrušen předcházejícím znakem procenta. Zvolila jsem proto způsob procházení regulárních výrazů znak po znaku. V posledním kroku se ještě pomocí funkce `str_replace` nahradí speciální symboly takovou sekvencí znaků, která ruší jejich jiný význam.

NQS

Součástí této části implementace je i řešení rozšíření NQS. Při procházení regulárních výrazů a nalezení některého z kvantifikátorů (+, *) se zkontroluje, jaký znak je následující. Pokud se jedná opět o kvantifikátor, je z regulárního výrazu vynechán první (v případě prvního kvantifikátoru +), nebo druhý (v případě prvního kvantifikátoru *) operátor.

Vyhledání podřetězců

Na vyhledání nejdelšího podřetězce, který odpovídá určitému regulárnímu výrazu, jsem použila funkci `preg_match_all`, která do trojrozměrného pole uloží řetězec, který byl ve vstupním textu nalezen, a pozici, na kolikátém znaku daný řetězec začíná. Pro potřeby vládání konce formátovacích příkazů jsem toto pole rozšířila o další prvek, a to pozici konce řetězce, kterou lze spočítat jako součet délky řetězce a pozice počátku. Jelikož ve výsledném textu nemáme formátovat prázdné řetězce, celé pole se poté projde a odstraní se ty prvky, které obsahují v buňce nalezeného textu prázdný řetězec.

Přidání formátovacích příkazů

Nejprve je vyhledán dosud nezpracovaný začátek či konec řetězce, před (respektive za) který se má formátovací příkaz vložit, s nejnižší hodnotou. Poté je vybrána a vložena formátovací značka a pozice všech dosud nepřidaných formátovacích značek jsou zvýšeny o délku právě vloženého řetězce. Zrovna použitá pozice formátovací značky je přepsána na nekonečno, aby bylo možno snadno odlišit již vložené a ještě nepoužité značky. Tato část se opakuje, dokud nejsou vloženy všechny formátovací značky.

Závěr

V posledním kroku se zkontroluje, zda byl zadán parametr --br a případně se nahradí všechny znaky konce řádku kombinací příslušné formátovací značky a znaku konce řádku. Jako poslední je zavolána funkce `file_put_contents`, která otevře a přepíše, nebo vytvoří výstupní soubor a zapíše do něj (případně na standardní výstup) výsledný text.