Dokumentace úlohy CHA: C Header Analysis v PHP5 do IPP 2016/2017
Jméno a příjmení: Petr Jůda
Login: xjudap00

## Rozbor zadání:

Cílem projektu bylo vytvořit php script, který na svém vstupu přijímá hlavičkové soubory zdrojového kódu a na výstupu generuje XML strukturu nalezených funkcí. Po nastudovaní zadání, jsem se kvůli nedostatečné znalosti OOP, rozhodl pro strukturovaný přístup programování. Výsledný program jsem rozdělil do několika logických bloků popsaných níže.

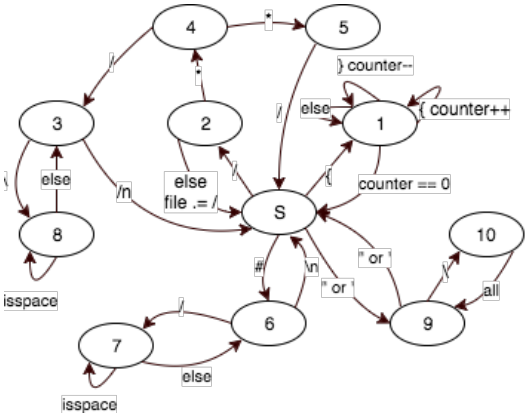
## Zpracování parametrů:

Script je možné spustit s několika na sobě nezávislými parametry, proto celý proces začíná validací těchto parametrů, případně nastavením jejich výchozích hodnot. Validace je prováděna ve dvou krocích. Nejprve je pomocí nastavených pravidel funkce `getopt()` získáno pole, které obsahuje hodnoty zadaných parametrů. Následně je volána funkce `check_args()`, která kontroluje, zda jsou zadané hodnoty přípustné. Také nastaví výchozí hodnoty pro všechny, tedy i nezadané parametry.

## Načtení a vyfiltrování souboru:

Pokud byl na vstupu zadán parametr –input=soubor celý proces filtrování je spuštěn pouze nad jedním souborem. Pokud byla na vstupu zadána složka, je pomocí rekurzivního průchodu a regulárního výrazu proces postupně spouštěn nad všemi hlavičkovými soubory. Filtrování začíná voláním funkce `file_extract()`, která načte celý obsah souboru do proměnné. Následně je voláním funkce `clear_rubbish()` spuštěna lexikální analýza, která ze souboru odstraní všechny komentáře, makra, textové řetězce a ze všech definicí funkcí udělá deklarace. Tímto je ošetřena práce s více bytovými znaky, které se mohli vyskytovat v komentářích a dále lze využívat obyčejné funkce na zpracování textu.

## Schéma konečného automatu lexikální analýzy:




## Získání a zpracování jednotlivých funkcí:

Ze zbylého textu, je pomocí regulárního výrazu, který definuje kostru funkce, získáno pole obsahující jednotlivě oddělené deklarace funkcí. Nad každou položkou pole je volána funkce `parse_function()`, která pomocí regulárních výrazů a vestavěných funkcí `(l/r)trim()`, `explode()` rozdělí funkci na jméno, návratový typ a obdobným způsobem zpracuje všechny její parametry. Zároveň je postupně procházeno pole vstupních parametrů programu a je upravována proměnná `$can_write`, která ovlivňuje, zda bude bude funkce obsažena ve výsledném XML souboru.

## Generování XML:

Díky dobré dekompozici zadaní, bylo možné výsledný XML soubor generovat pomocí zápisu do souboru již během postupného zpracování, bez využití xml knihovny. Pro zápis elementu do XML je volána funkce `print_xml()`, která mimo jiné přijímá vypisovaný řetězec a značku `$format`, který specifikuje, zda se má výstup odřádkovat. Počet tisknutých mezer ovlivňuje globální proměnná `$space_count`. Pokud byl zadán –output=file, je výsledný řetězec zapsán do souboru, jinak je pro zápis otevřen `php://stdout`. Toto řešení je možné díky tomu, že standartní výstup není skriptem nijak jinak využíván.