Dokumentace úlohy CHA: C Header Analysis v PHP 5 do IPP 2016/2017
Jméno a příjmení: Jakub Handzuš
Login: xhandz00

# 1 Zadání

Úkolem skriptu je analyzovat hlavičkové soubory jazyka C, které jsou napsány podle standardu ISO C99 v kódování UTF-8 a následně uložit zpracovaná data do XML formátu. Skript je napsán v jazyce PHP 5, přičemž je možné jej rozčlenit do několika funkčních celků, které budou blíže popsány v ostatních částech dokumentace.

# 2 Popis řešení úkolu

## 2.1 Zpracování vstupních argumentů

Prvním úkolem skriptu je správně analyzovat vstupní argumenty, jejichž zpracování je implementováno v samostatné funkci `argumentsLoad()`, která kontroluje shodu s předem specifikovanou množinou argumentů. Pokud se nenajde žádná shoda, skript vyhodnotí argument jako neplatný a je ukončen se zprávou vypsanou na chybový výstup, kde je blíže popsána konkrétní chyba. Stejně je skript ukončen, pokud nastane nesprávná kombinace nebo se argumenty vyskytují duplicitně. Při argumentech, které mohly obsahovat číselnou hodnotu, se provádí kontrola správnosti hodnoty. V případě bezproblémového načtení jsou nastaveny příznaky, které určují následující chování skriptu.

## 2.2 Zpracování adresáře

Ve vstupních parametrech se může vyskytnout místo konkrétního souboru i celý adresář. Pro tento případ je v skriptu implementována funkce `getHeadersFile()`, která má na starost prohledávání konkrétního adresáře včetně jeho podadresářů a vyhledávání hlavičkových souborů. Princip prohledávání podadresářů je založen na rekurzívním volání, přičemž funkce vrací pole cest souborů, které je ještě následně seřazeno pomocí další funkce `sortFiles()`. Tato funguje na principu výměny klíče a hodnoty v poli, kde se do klíče uloží cesta a do hodnoty se vypočte hloubka zanoření na základě počtu lomítek v cestě. Toto pole se seřadí pomocí vestavěné funkce a následně se znovu nahraje klíč položky do její hodnoty. Tímto způsobem je zajištěno, aby byly soubory s menším zanořením analyzovány dříve než ty s větším, protože skript v případě adresáře analyzuje každý soubor zvlášť, a to podle pořadí, v jakém jsou v poli souborů.

## 2.3 Úprava obsahu vstupního souboru

Před samotnou analýzou hlavičkového souboru jazyka C musí skript odstranit části kódu, konkrétně komentáře, makra a řetězce, jelikož mohou způsobovat nesprávné chování programu při pozdější analýze. Odstraňování je založeno na principu konečného automatu, jehož použití bylo vhodnější než regulární výraz. Tento automat přijímá znak po znaku ze vstupního souboru, který byl ještě před tím rozdělen na pole znaků, z důvodu kompatibility přijímání celé abecedy UTF-8 kódování.

Na obrázku 1 je znázorněn zjednodušený konečný automat, který se snaží co nejpřesněji znázornit vnitřní implementaci funkce `junkDelete()`, aniž by musely být stavy duplikovány. V případě aktuálního znaku `\` automat nepřechází do dalšího stavu, ale nastavuje příznak `ESCskip`, který způsobí přeskočení následujícího znaku. Přechod ze stavu `MACRO` do stavu `SLASH` je znázorněn přerušovaně z toho důvodu, že pokud další příchozí znak nebude `*` nebo `/`, tak se automat vrátí zpět do stavu `MACRO` (znázorněno pomocí + `mFlag`). V případě konce víceřádkového komentáře, který začal v makru, se automat nevrací do stavu `SAVE`, ale do stavu `MACRO`. Automat si při přechodu do stavu `SAVE` ukládá počáteční pořadové číslo znaku a při opuštění tohoto stavu se úsek od počáteční pozice do aktuální zkopíruje do výsledného textu. Tímto způsobem zajistíme ukládání pouze toho textu, který je potřebný pro další analýzu.

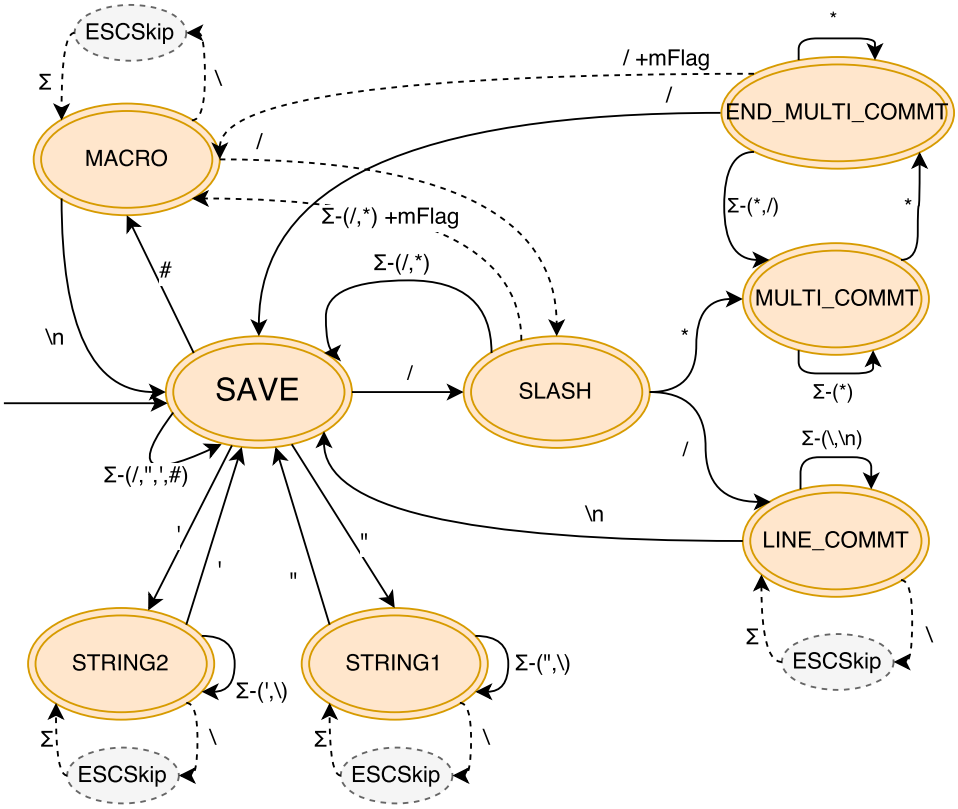
## 2.4 Identifikace funkce a analýza informací

Po odstranění všech nepotřebných znaků se tento upravený text předá funkci `analysis()`, která se stará o nalezení deklarace či definice funkce. Princip hledání je založen na regulárním výrazu, který hledá definovanou posloupnost znaků. Samotný výraz analyzuje definici nebo deklaraci na název, parametry a návratový typ funkce, kterou následně ukládá do pole polí, podle zmíněných kategorií. Následně je pole argumentů analyzováno funkcí `realPar()`, která spočítá počet sledovaných parametrů, odstraní prvky pole, které nejsou sledovány, případně identifikuje, zda má daná funkce proměnlivý počet parametrů. Tato funkce také vygeneruje XML řetězec parametrů, který se na základě přepínače `-max-par` rozhodne, zda se informace o funkci zapíší, nebo

zahodí. Ve funkci `analysis()` se vyhodnocují i další přepínače. V případě přepínače `-no-inline` se funkce, které obsahují slovo `inline` v návratové hodnotě, přeskočí. Pokud byl zadán přepínač `-no-duplicates`, skript si ukládá názvy analyzovaných funkcí a v případě shody je analýza dané funkce přeskočena.

## 2.5 Zápis XML elementů do výsledného souboru

Generování XML dokumentu je realizováno pomocí konkatenace řetězců, které si funkce mezi sebou posílají. V případě přepínače `-pretty-xml` jsou před každý řádek přidány znaky odsazení a na konci znak nového řádku. Zápis do výsledného souboru, připadne na standardní výstup, se provádí až na samém konci programu.




Obrázek 1: Konečný automat