Dokumentace úlohy CHA: C Header Analysis v Perlu do IPP 2011/2012
Jméno a příjmení: Petr Dvořáček
Login: xdvora0n
E-mail: xdvora0n@stud.fit.vutbr.cz

1 Úvod

Tento dokument pojednává o implementaci projektu do předmětu IPP. Myšlenkou tohoto úkolu bylo vytvořit skript na jednoduchou analýzu hlavičkových souboru jazyka C. V tomto projektu jsem použil pouze knihovnu `File::Find`, ve které se nachází funkce `find`, která hledá soubory v adresářové hiarchii. Žádné další, podle mého názoru, nebylo potřeba.

2 Zpracování parametrů

Nejdříve si nastavím řídicí proměnné. Samotné zpracování parametrů probíhá ve smyčce `foreach $i (0 .. $#ARGV)` za pomocí regulárních výrazů. Tam také kontroluji zda některé parametry nebyly zadány vícekrát. Nekompaktibilitu parametrů jsem řešil pouze pro tisk nápovědy. Například `$ ./cha.pl --help --pretty-xml=42` vytiskne chybovou hlášku. Jde o nesprávně zadané parametry.

Použití knihovní funkce `Getopts::long` mně přišlo v tomto případě zdlouhavé.

3 Vyhledání souborů

Po zpracování parametrů následuje vyhledání souborů. Nejdříve zkontroluji zda zadaná cesta existuje. Pak pomocí subrutiny `find` vložím do pole `$header_files` všechny názvy hlavičkových souborů nacházející se v zadané cestě.

4 Analýza funkcí

Každý soubor z `$header_files` analyzuji jednotlivě. Nejdříve soubor otevřu a uložím jej celý do řetězce `$temp_file`, se kterým pak pracuji. Z tohoto řetězce odeberu komentáře a makra. Poté na něj aplikuji regularní výraz, který najde všechny funkce. U každé z nich mi uloží do pole `$parsed_functions` typ návratvé hodnoty, název a řetězec všech jejich parametrů. Tyto položky pak zpracovávám a vybírám funkce tak, aby vyhovovaly volaným parametrům skriptu. Vyhovující funkce pak ukládám do trojrozměrného pole `$function`, ze kterého následně generuji kód.

4.1 Nalezení funkce

```
[};]\s*([\w\*][\w\s\*]*)\s(\w+)\s*\(([,\w\s\*\.]*)\)[\s]*(?=[;{])
```

Nalezení funkcí probíhá za pomocí regularního výrazu uvedeného výše. Pomocí něho uložím do pole `$parsed_functions` její parametry. Z mých zkušeností se regulární výrazy dobře píší, ale špatně čtou, proto se o něm trošku rozepíšu.

Reg. výraz	**Popis**
`[};]\s*`	Konec předchozího příkazu. Následováno libovolným počtem bílých znaků.
`([\w\*][\w\s\*]*)`	Návratový typ funkce, který se uloží do pole.
`\s`	Mezi návratovým typem a názvem je vždy mezera.
`(\w+)\s*`	Název funkce, který se uloží do pole. Následováno libovolným počtem bílých znaků.
`\(`	Začátek parametrů funkce.
`([,\.\w\s\*]*)`	Řetězec paramterů funkce, uloží se do pole.
`\)\s*`	Konec parametrů funkce.
`(?=[;{])`	Taková malá „pojistka“. Zda to opravdu pokračuje závorkou, či středníkem.

Tento regulární výraz však nenalezne funkce nacházející se na začátku souboru. Právě kvůli první části, kde testuji středník či konec závorky. Tento problém jsem vyřešil tím, že do proměnné `$temp_file` vložím středník před čtením souboru.

4.2 Zpracování funkcí

Jednotlivé funkce jsou pak uloženy za sebou v poli `$parsed_functions`. Pole vypadá následovně: (pro první funkci) návratová hodnota, název, parametry, (pro druhou funkci) návratová hodnota, název, parametry atd. Proto jsem použil počítadlo `$i`, díky kterému zjišťuji, na jaké položce se zrovna nacházím. Z každé položky odstraním bílé znaky nacházející se na začátku nebo na konci řetězce. Řetězec parametrů rozdělím na jednotlivé parametry pomocí vestavěné funkce Perlu `split()`.

Podle nastavení argumentů programu, se rozhodne zda se nalezená funkce uloží či ne. O tom rozhoduje řídicí proměnná `$ok`.

4.3 Uložení funkce

Funkce ukládám do trojrozměrného pole, kde první index (`$id_fce`) značí pořadí funkce. Ten pak po uložení inkrementuji.

Proměnná	**Popis**
`$function[$id_fce][0][0]`	Název souboru
`$function[$id_fce][1][0]`	Název funkce
`$function[$id_fce][2][0]`	Proměnný počet argumentů
`$function[$id_fce][3][0]`	Návratový typ funkce
`$function[$id_fce][4][0]`	Celkový počet parametrů
`$function[$id_fce][5][$j]`	Jednotlivé parametry, kde $j = 1, 2, 3, \ldots, n$ značí jejich pořadí.

5 Generování XML souboru

Podle parametru `--xml-prety` nastavím pomocné proměnné, které generovaný XML soubor zpřehlední. Pak generuji kód za pomocí pole `$function` buď na standardní výstup, nebo do souboru.

6 Závěr

Tento skript by měl být funkční pro jazyk C podle standardu ISO C99. Měl by fungovat na většinu „jednoduchých“ funkcí.