**Dokumentace úlohy CHA: C Header Analysis v Python 3 do IPP 2016/2017**

Jméno a příjmení: **Petr Buchal**

Login: **xbucha02**

**Základní popis**

Skript slouží k analýze hlavičkových souborů (přípona .h) jazyka C (norma ISO C99). Konkrétně analyzuje deklarace funkcí. Skript je psaný v jazyce Python.

Vstup: Na vstupu skript očekává buď adresu hlavičkového souboru jazyka C a nebo adresu složky, kterou má projít a nalézt v ní tyto soubory. Vstup je specifikován parametrem --input. Pokud je skript spuštěn bez tohoto parametru, čte skript z adresáře, ve kterém se nachází.

Výstup: Na výstup skript tiskne XML soubor obsahující informace o funkcích v procházených souborech/souboru. Tisk probíhá implicitně na standartní výstup. Při zadání parametru --output tiskne skript do souboru, jehož název a umístění je určeno tímto parametrem.

Parametry skriptu: Skript může být spuštěn celkem s osmi různými parametry. Parametr --help vypíše nápovědu. Parametry --input a --output jsou využívány jako ukazatele z jakého souboru/složky má skript data číst a do jakého souboru je má ukládat. Parametr --pretty-xml udává, jaké bude odsazení jednotlivých nodů, může být zadán s hodnotou i bez hodnoty. Pokud je parametr zadán bez hodnoty, odsazení nodů bude implicitně 4. Pokud je zadán parametr --no-inline, skript nebude zkoumat funkce, které jsou deklarované s inline specifikátorem. Při zadání parametru --max-par se budou ignorovat funkce, které mají více parametrů než je hodnota --max-par. Parametr --no-duplicates říká, že se nebudou zkoumat funkce se stejným jménem, jaké už je uloženo v databázi procházených funkcí. Parametr --remove-whitespace odstraní přebytečné bílé znaky.

**Implementace**

Třídy: Skript obsahuje tři třídy (database, function a parserForFile). Třída database má dvě proměnné, proměnnou functions, která je pole (obsahuje instance třídy function), které slouží k ukládání analyzovaných funkcí a proměnnou parameters (rovněž pole), která slouží k uložení parametrů skriptu pro rychlý přístup k nim. Třída function má pět proměnných, první je řetězec name (jedná se o jméno funkce), druhá je pole parameters (slouží k ukládání parametrů funkce), třetí je řetězec file (slouží k uložení jména souboru, popřípadě jména a cesty k souboru, ve kterém se funkce nachází), čtvrtá je řetězec rettype (obsahuje návratový typ funkce) a pátá je řetězec varargs (obsahuje řetězec „yes“ nebo „no“ podle toho, zdali se jedná o funkci s proměnným počtem argumentů). Třída parserForFile obsahuje pouze metody pro analyzování souborů.

Hlavní tělo programu: Program nejdříve pomocí několika řádků kódu zpracuje argumenty (využívá se knihovny argparse), pokud nastává chyba je ukončen s návratovou hodnotou 1. Následně se kontroluje existence a práva cest, které jsou udávány argumenty --input a --output. Pokud jsou parametry skriptu validní a cesty korektní, vytvoří se instance třídy database. Následně probíhá buď rekurzivní procházení adresářů a jejich souborů pomocí metody recursive\_gold (pokud je parametrem --input zadána složka nebo nebyl parametr zadán vůbec) a nebo se pomocí metody analysa přímo analyzuje soubor (--input obsahuje adresu souboru). Po doběhnutí analýzy souboru/souborů se pomocí metody printDatabase vytiskne požadovaný XML soubor v požadovaném formátování. Jednotlivé části skriptu jsou popsány v následujících řádcích podrobněji.

Rekurzivní průchod adresáři:

Rekurzivní průchod adresářovou strukturou umožňuje metoda recursive\_gold. V každé složce tato metoda kontroluje její položky (zdali se jedná o soubor nebo o složku). Pokud se jedná o složku, volá se nad ní opět recursive\_gold, jedná-li se o soubor, kontroluje se přítomnost koncovky „.h“. Pokud je soubor hlavičkový, volá se nad ním metoda analysa, která analyzuje jeho obsah.

Analýza souboru:

Analýza souboru začíná v metodě analysa. Tato metoda volá metodu readByChar ze třídy parserForFile, která už daný soubor přímo analyzuje. V této třídě je rovněž metoda whiteSpaceStrech, která slouží ke změně jiných bílých znaků než mezer na mezery (využívá se v okamžiku, kdy je specifikátor složený z více slov). V readByChar se využívá několik značek (inFunction, inComment, inString, inMacro, inBody), které analyzátoru říkají, v jaké oblasti souboru se právě nachází (vždy může být aktivní pouze jedna značka). Pokud je aktivní značka inFunction metoda zkoumá deklaraci nějaké funkce, po ukončení zkoumání deklarace funkce se instance třídy function (v té jsou uloženy informace o dané funkci) rovnou vkládá do databáze a pokračuje se v dalším průchodu souborem.

Tisk XML souboru: Pro tisk XML souboru s obsahem databáze se volá metoda printDatabase. Ta konkatenuje informace o jednotlivých funkcích do jednoho řetězce, který je poté předán metodě output\_func. Tato metoda se podle parametru --output rozhodne, kam bude daný řetězec tiknout a vytiskne ho. Poté se skript ukončí.

Závěr: Skript byl otestován jak poskytnutými, tak přídavnými testy. K porovnání XML souborů jsem využil nástroj JExamXML. Testování proběhlo na školním serveru Merlin.