Dokumentace úlohy CHA: C HeaderAnalysis v Python3 do IPP 2014/2015
Jméno a příjmení: Sára Škutová
Login: xskuto00

1. Zadání

Vytvořit program (skript) v jazyce Python3 pro analýzu hlaviček funkcí jazyka C.

2. Implementace

2.1. Zpracováníparametrů

Parametry jsou zpracovávány pomocí `sys` modulu, seznamu `sys.argv` a cyklu. Před samotným zpracováváním se vymaže první prvek, čímž se lze zbavit názvu programu, dále následuje cyklus, který daný seznam prochází. V případě chyby (nedovolená kombinace, chybějící části parametrů, opakování parametrů…) se vypíše chybové hlášení a program se ukončí s daným návratovým kódem.

2.2. Práce se soubory

Program může pracovat buď s jedním vstupním souborem, nebo s adresářem. V případě, že bude v parametrech zadán adresář, začnou se od dané složky rekurzivně vyhledávat soubory s koncovkou .h. Jména souborů, která odpovídají formátu, se uloží do seznamu. Pokud vstupní parametr chybí, pracuje se s aktuálním adresářem. O rekurzivní vyhledávání souborů se stará funkce `get_file()`.

Výstupní soubor může být také zadán parametrem a v případě, že daný parametr chybí, tak se data vypíší na standardní výstup.

2.3. Podpůrné funkce pro analýzu

Analýza hlavičkových souborů potřebuje pro svou správnou činnost pozměněný text (odstraněné komentáře, makra atd.), a proto se před každou analýzou volají potřebné podpůrné funkce. Většina podpůrných funkcí byla převzata z mého předchozího CST projektu. Před samotným zpracováním se vždy daný soubor otevře a celý jeho obsah se uloží do proměnné, následně se soubor uzavře a již se s ním nepracuje.

2.3.1. Odstranění maker

Tuto operaci provede funkce `DESTROY_Macro()`. V této funkci se v cyklu prochází jednotlivé řádky a pomocí stavového automatu se rozhoduje, zda daný řádek uložit do výsledného textu nebo ne. Podporují se i makra, jenž pomocí znaku \ pokračují na dalším řádku.

2.3.2. Odstranění komentářů

Funkce `DESTROY_Comment()` odstraní veškeré komentáře, ale zachová přitom řetězce, které mohou jako komentáře vypadat. Text se prochází po jednotlivých znacích a pomocí stavového automatu se rozhoduje, zda dané písmeno vložit do výsledného textu nebo ne. Řádkové komentáře označené `//`, jenž pomocí znaku \ pokračují na dalším řádku, jsou taktéž podporovány.

2.3.3. Odstranění řetězců

Provádí funkce `DESTROY_String()`. Odstraní řetězce a znakové literály, komentáře již budou odstraněny, takže se nemusí řešit „řetězce“ v komentářích. V této funkci se taktéž pomocí stavového automatu rozhoduje, zda daný znak bude ve výsledném textu či nikoliv.

2.3.4. Odstranění uživatelsky definovaných typů a bloků

Regulární výraz odstraní řádky, na kterých se vyskytují deklarace uživatelských typů. Pokud se ale daný uživatelský typ vyskytuje v deklaraci funkce, tak se daný řádek přidá do výsledného textu. Dále se ve funkci `DESTROY_Blocks()` pomocí konečného automatu odstraní veškeré bloky, čímž by měly v daném textu zůstat pouze hlavičky funkcí.

2.4. Analýza hlaviček funkcí

Poté co se text upraví do podoby pro analýzu, tak se v cyklu prochází jednotlivé hlavičky. Jejich analýza je rozdělena na dvě části. V první se analyzuje jméno a návratový typ funkce, ve druhé části pak počet jednotlivých parametrů a jejich datové typy.

2.4.1. Jméno funkce a návratový typ

Jméno a návratový typ funkce se z hlavičky získá pomocí cyklu, kdy se čte po znacích, dokud se nenarazí na levou závorku (.

Pokud uživatel zadá, že nechce vyhledávat funkce ze specifikátorem inline, tak se provede kontrola, zda daná hlavička toto klíčové slovo neobsahuje. Pokud ano, začne zpracovávat další funkce. Dále může uživatel zamezit výpis duplicitních funkcí. Názvy jednotlivých funkcí se ukládají do pomocného seznamu. Pokud tam již daný název je, tak se aktuálně rozpracovaná hlavička zahodí a začne se zpracovávat nová.

Po té co se podaří oddělit návratový typ a jméno funkce (najde se jméno funkce a následně se smaže), tak se provedou dodatečné úpravy týkající se mezer v návratovém typu (pokud byl zadán daný parametr), následně se pokračuje v analýze parametrů funkce.

2.4.2. Parametry funkce

Parametry se také získají pomocí cyklu, kdy se do pomocné proměnné načtou všechny znaky, které se nachází mezi závorkami.

Následně se kontroluje, zda je to funkce bez parametrů nebo zda nějaké parametry má. Pokud se jedná o funkci bez parametrů (včetně parametru void), přeskočí se celá následující analýza a připraví se výpis. V opačném případě se text rozdělí na jednotlivé parametry a jejich seznam se následně prochází.

V případě, že se najde parametr, který indikuje proměnný počet parametrů (…), tak se nastaví proměnná `varargs` a dále se pak již s daným parametrem nepracuje.

Po té co se podaří oddělit datový typ a identifikátor parametru (najde se identifikátor a následně se smaže), tak se provedou dodatečné úpravy týkající se mezer v datovém typu (pokud byl zadán daný parametr). Následně se datový typ uloží do pomocného seznamu (získá se tím pořadí, v jakém se parametry v hlavičce vyskytují). Na konec se ještě kontroluje, zda daná hlavička nemá více parametrů, než bylo povoleno (dle vstupních parametrů). Pokud má, tak se daná hlavička nevypisuje.

2.5. Formátování výstupu

Na výstup se vypíše vždy celý obsah proměnné `vystup`. Při spuštění skriptu se do dané proměnné nahraje požadovaná hlavička XML. Dále se provede funkce `xml_hlava()`, která vypíše hlavičku `<functions>` a dosadí patřičnou hodnotu k elementu `dir`. Po té co se provede analýza hlavičky, tak se dodatečně pomocí funkce `correct_files()` převede název souboru, ve kterém byla hlavička nalezena na relativní vzhledem k hodnotě v elementu `dir`. Následně se do proměnné `vystup` přidá položka `<function>` i s požadovanými elementy. Pokud v souboru nebyla žádná hlavička, tak se `<function>` nevypíše. Dále se v cyklu vypíší nalezené parametry. V této i v předchozí fázi výpisu se vždy bere v potaz existence parametru –pretty-xml, kdy se musí dodržovat potřebné odsazení. Manipulaci s proměnnou `vystup` ukončí funkce `xml_pata()`, která uzavře hlavičku `</functions>`. Následně se daná proměnná vypíše do požadovaného souboru.