**Dokumentace úlohy CHA:** C Header Analysis v Pythonu do IPP 2015/2016  
**Jméno a příjmení:** David Kolečkář  
**Login:** xkolec07

Úkolem projektu bylo vytvořit skript v jazyce Python, pro analýzu hlavičkových souborů jazyka C, který vytvoří databázi nalezených funkcí. Skript je napsán procedurálním stylem a dělí se na několik částí: zpracování argumentů, načtení vstupních dat, zpracování hlavičkových souborů, tisk chybových hlášení. Všechny uvedené části jsou popsány níže.

**Zpracování argumentů**

Argumenty se zpracovávají ve funkci argument(), ihned po spuštění skriptu. Ke zpracování argumentů je využívána knihovna argparse. Zároveň se zde kontrolují nesprávné kombinace či nepřípustné hodnoty argumentů. V případě chyby se vrací návratový kód 1. Při zadání přepínače --help se vytiskne nápověda na standardní výstup.

**Načtení vstupních dat**

Načtení vstupních dat probíhá ve funkci godir(), která rekurzivně prohledává adresářovou strukturu. Funkce je volána, pokud je zadán parametr --input s názvem adresáře, nebo parametr --input není zadán vůbec (prohledává se aktuální adresář a všechny jeho podadresáře). Pokud funkce najde adresář tak se zanoří, pokud najde hlavičkový soubor, uloží cestu k souboru a jeho název do seznamu.

**Zpracování hlavičkových souborů**

Ke zpracování jednotlivých hlavičkových souborů slouží funkce detection(). Funkce otevře hlavičkový soubor a uloží obsah souboru do proměnné. Poté pomocí regulárních výrazů jsou odstraněny řádkové komentáře a definice maker. K odstranění více řádkových komentářů a těl funkcí slouží konečný automat se třemi stavy, který po znacích prochází obsah souboru.

Dále rozdělíme jednotlivé funkce z upraveného souboru na seznam funkcí pomocí split() podle středníku. Iterujeme přes seznam funkcí a zjišťujeme název funkce a návratovou hodnotu funkce. Pro každou funkci kontrolujeme, zda je definována jako inline (v případě že je zadán argument --no-inline jsou tyto funkce přeskočeny), zda obsahuje proměnný počet parametrů (tento údaj se zaznamenává do atributu vararg). K ověření jestli se v souboru nevyskytuje více funkcí se stejným jménem, je vytvořen seznam s názvy funkcí. A v případě zadaní argumentu –-no-duplicates, se při každé iteraci kontroluje, zda již název není obsažen v seznamu, pokud ano je funkce přeskočena. Argument --max-par=n bere v úvahu pouze funkce, které mají n čí méně parametrů. Pokud je zadán argument --remove-whitespace jsou odstraněny přebytečný mezery z návratových typů funkcí. Před zpracováním parametrů funkce se vypíše element function s atributy soubor ve kterém byla funkce nalezena, název funkce, návratový typ funkce a jestli se jedná o funkci s proměnným počtem parametrů.

Parametry se rozdělí pomocí split() podle čárky do seznamu. Iterujeme přes seznam parametrů a zjistíme typ parametru. Evidujeme také číslo parametru, a pokud je zadán argument skriptu --remove-whitespace odstraníme přebytečné mezery z typu parametru. Nakonec se vypíší elementy param obsahující číslo parametru a jeho datový typ, v případě zadaní argumentu --prety-xml je navíc před element vytisknuto odsazení a za element znak nového řádku.

**Tisk chybových hlášení**

Tisk chybových hlášení je implementován ve funkci printError(), která vypíše na standardní chybový výstup chybovou hlášku a ukončí běh skriptu daným návratovým kódem.

**Závěr**

Skript byl testován pomocí poskytnutých testů na školním serveru Merlin s operačním systémem CentOS a na operačním systému Ubuntu 14.04. Všechny testy proběhly úspěšně včetně rozšíření PAR.