Implementační dokumentace k 1. úloze do IPP 2023/2024
Jméno a příjmení: Adam Juliš
Login: xjulis00

Zvolení návrhového vzoru
První autorova zkušenost s návrhem a následnou implementací objektově orientovaného programováním. Snaha byla dodržet Command – Příkazový návrhový vzor. V potaz se bral i vzor interpret. Následuje popis celého programu:

Funkce main
Pokud funkce proběhne, návratová hodnota programu je 0.
Nejprve se volá (jediná) funkce `check_args`, kontroluje validitu argumentů.
Již zde může dojít k přerušení běhu programu (k tomu může dojít na více místech), vždy shodným způsobem – metodou `exit_error` třídy `ErrorCode` (díky vlastnímu Enum slovníku lze jednoduše přidávat další chybové kódy) navrátí korektní chybovou hodnotu programu a vypíše hlášku na standartní chybový výstup.
Následně se vytvoří objekt `context` třídy `Context`, která načte ze standartního vstupu data a zároveň nad nimi zavolá vnitřní metodu k odstranění nadbytečných bílých znaků a komentářů (v `__init__` volá vnitřní metodu `trim_white_space`). Tento objekt poté slouží k vytvoření objeku `parser` třídy `Parser`.
Samotná analýza probíhá zavoláním metody `parse()` zmíněného objektu.
Pokud analýza proběhne v pořádku, vytiskne se na standartní výstup xml reprezentace vstupního kódu, tato část je řešena neobjektově čtyřmi řádky kódu.

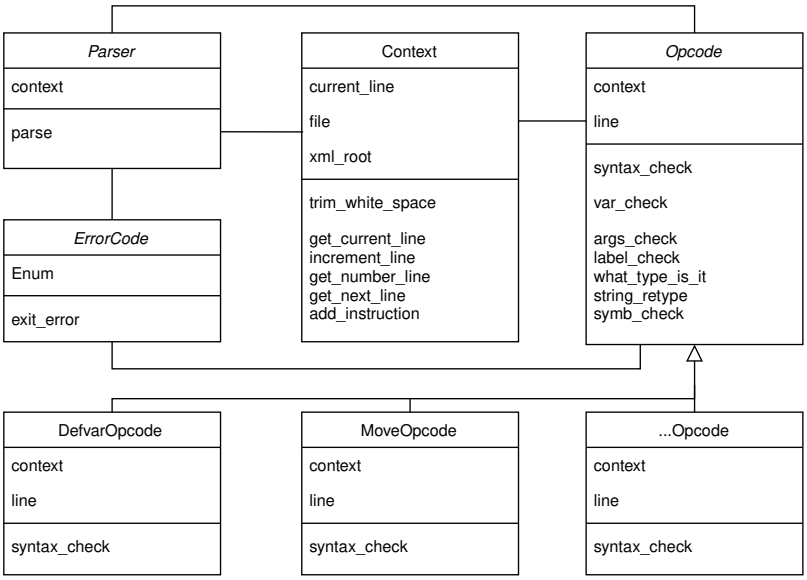
Třída Parser
Obsahuje jedinou metodu `parse()`. Princip spočívá v cyklickém procházení jednotlivých řádků načteného kódu v třídě `Context`, to obstarávají její metody (zejména `increment_line`, `get_next_line`). Nejprve kontroluje hlavičku kódu a pak jednotlivé příkazy. Využívá slovník `opcode_enum` pro detekci příkazu i pro zvolení vhodné podtřídy třídy `Opcode` pro analýzu daného řádku. Tento proces se opakuje do konce souboru nebo ukončení chybovým kódem. Využívá try-except pro detekci 1 slovní instrukce. Při snaze elegantní opravy chybného chybového kódu při výskytu dvou hlaviček po sobě kód obsahu o jednu podmínku v cyklu více.

Třída Opcode a její podtřídy
Rodičovská třída všech operandů (každý operand má svojí podtřídu), obsahuje metody pro kontrolu syntaxe jednotlivých argumentů operandů. Pro detekci jednotlivých možných operandů (metody `var_check`, `label_check`, `string_check` a `symb_check`) se využívá pro přehlednost a jednoduchou modifikovatelnost další slovník, `regex_enum`. Jednotlivé operandy pak obsahují jedinou metodu, která je vhodně poskládaná z metod rodiče (volení pořadí metod a jejich argumentů podle příkazu, který se má kotrolovat. Poslední volaná metoda `add_instruction` se provádí nad objektem `context`, tím dojde k zapsání instrukce do xml stromu.
Autor zde vidí, že mohl přistupovat jednodušeji a nevytvářet tolik podobných samostatných potříd, místo toho tvořit podtřídy podle pravidel vyžadujících operandů.

Třída Context
Zmíněna včetně metod již v předchozích bodech. Obstarává tedy obsluhu nad vstupními daty a tvorbu xml stromu.

Třída ErrorCode
Zmíněna již v popisu funkce main.
Vyvolávána je z tříd `Parser` a `Opcode`.

Rozšíření
NVP
- více napsáno v prvním bodě dokumentace.



UML diagram tříd