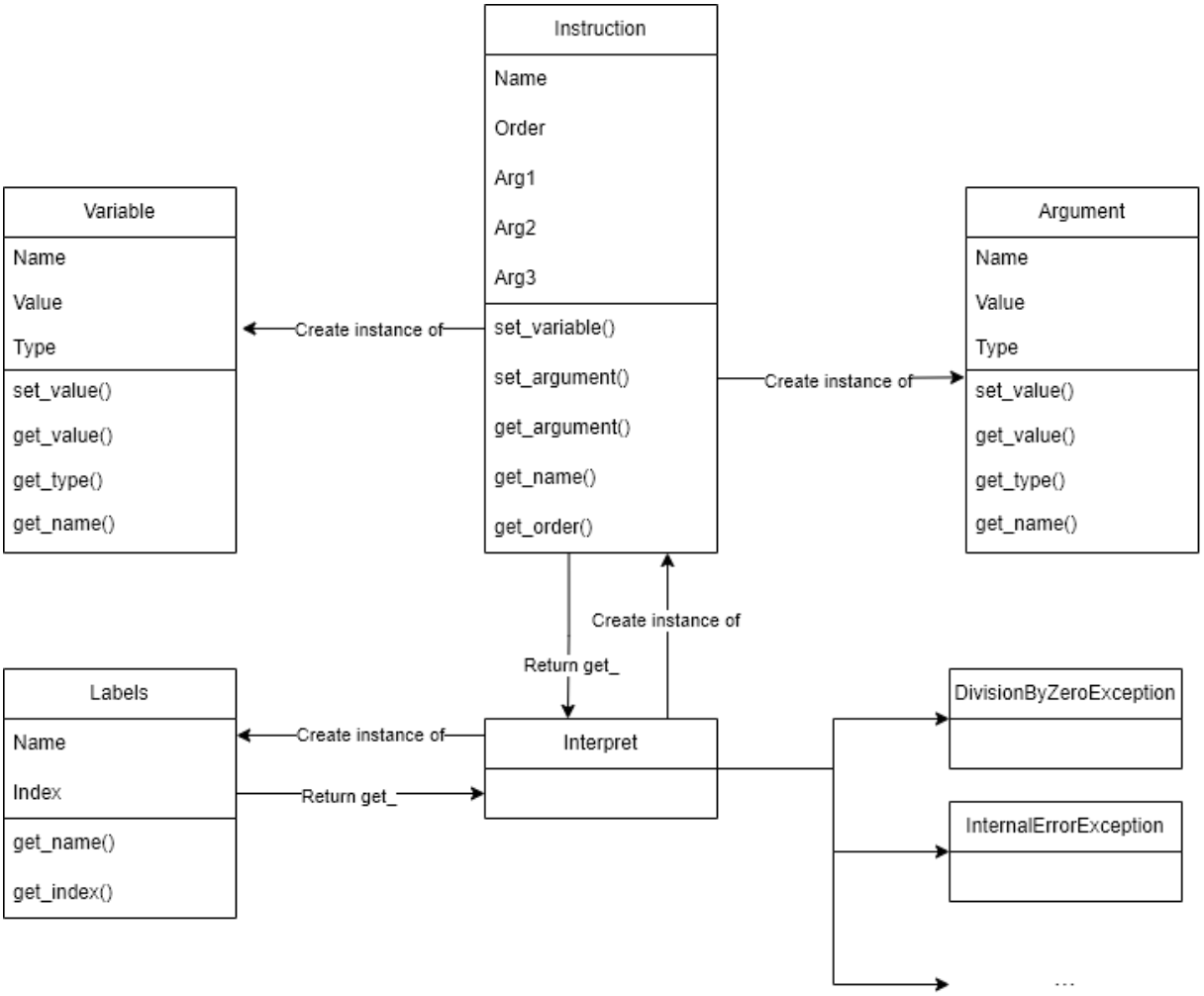
Implementační dokumentace k 2. úloze do IPP 2023/2024
Jméno a příjmení: Dalibor Kalina
Login: xkalin16

Importovaný XML soubor je nejprve rozdělen do bloků dle `<instruction>` bloků. Z tohoto bloku se vyjme název instrukce a její ordinální hodnota. Je vytvořena instance třídy `Instruction` s jejím názvem odpovídající názvu instrukce vyjmuté z bloku. Každá instance této třídy obsahuje mimo jiné 3 proměnné pro argumenty. Obdobně jako u rozdělení XML soboru do bloků, dojde nyní k rozdělení bloku instrukce na menší bloky s argumenty. Z těchto podbloků se vyjme hodnota a datový typ. Každý argument je pak uložen do instance třídy `Instruction` pomocí `set_argument()` nebo `set_variable()` v závislosti na tom, zda se jedná o argument typu proměnná (specifikované pomocí `type="var"`). `Variable` i `Argument` jsou samy o sobě třídy. Ve výsledku mám tedy jednu instanci třídy `Instruction` s až třemi nenulovými proměnnými, které jsou instance tříd `Variable/Argument`. Instance jsou potom uloženy do pole všech instrukcí pro další zpracování.

Mým cílem bylo neimplementovat odvozené třídy pro každou instrukci IPPcode24, které by dědily od jedné hlavní společné třídy pro instrukce. Místo toho existuje v mém programu jen jediná třída pro instrukce, obsahující právě tři argumenty, z nichž některé mohou být nevyužity v závislosti na typu instrukce.

Pole instrukcí je procházeno po prvcích. Nejprve se vezme název instrukce pomocí `get_name()` a vykoná se příslušná rutina pro danou instrukci. Pokud instrukce obsahuje argument, kterým je proměnná, tak se kontroluje existence proměnné v rámci. Rámec je sám o sobě pole, do kterého je ukládána proměnná při definici a hodnota při změně stávající hodnoty. Neexistence proměnné v rámci (krom instrukce `DEFVAR`) a čtení neinicializované proměnné vyvolá příslušnou chybu.



Chybové kódy jsou implementovány jako několik tříd, každá ve vlastním souboru v `./Exceptions`. Tyto třídy dědí přímo z `IPPException v ipp-core`.

Při instrukci `LABEL` dochází k tvorbě instance třídy `Labels` a uložení jména návěští spolu s indexem vykonávané instrukce. Tato instance je potom uložena do pole všech návěští. Tím se zapříčiní, že při skokových instrukcích (za předpoklad, že skok má být proveden) se nemusí zpětně dohledávat pozice návěští v předchozích instrukcích, ale vezme se uložený index v poli návěští. Jelikož jsou instrukce procházeny pomocí for-cyklu, tak stačí přenastavit index procházení na index návěští.