IPP documentation parse.php

Documentation of project implementation for IPP 2022/2023

Name and surname: Dominik Truchly

Login: xtruch01

Parse.php is a PHP program that takes IPPcode23 source code as input and outputs it in XML format. The program verifies the syntax of the source code, and if the syntax is correct, the program converts the source code into XML. The program starts by checking if the input has a header ".IPPcode23" and then begins parsing the code.

I defined multiple functions that helped me checking the code for correct syntax:

- The **type_check** function checks whether a given argument has one of three possible types: "int", "bool", or "string". If the argument matches one of these types, the function outputs an XML tag **<arg>** containing the argument and its type. Otherwise, the function outputs an error message and exits the program.
- The **label_check** function checks whether a given label has a valid format, i.e., it contains only alphanumeric characters and some special characters, such as underscores, dashes, and dollar signs. If the label has an invalid format, the function outputs an error message and exits the program.
- The **not_variable_check** function checks whether a given argument has any other format than variable. If it was a variable the program outputs instruction with it's opcode, order and continued checking for arguments. If there wasn't a variable, the function eoutputs an error message and exits the program.
- The **regex_check** function checks whether a given argument has a valid format for a variable or a constant. If the argument is a variable, it must have a specific format (GF/LF/TF@label), while if it is a constant, it must have one of four formats: "int@value", "bool@true/false", "string@value", or "nil@nil". If the argument has a valid format, the function outputs an XML tag **<arg>** containing the argument and its type. Otherwise, the function outputs an error message and exits the program.
- The **arguments_count** function checks whether the number of arguments in a given line of code matches the expected number. If the numbers do not match, the function outputs an error message and exits the program.

The program first checks if there are any command-line arguments, and if there are, it checks if the argument is "--help". If the argument is "--help", the program outputs a help message and exits. If there are more than two command-line arguments, the program outputs an error message and exits.

Then the program starts reading the input line by line. If a line is empty or begins with a "#" character, the program skips it. If the input has not encountered a valid header (.IPPcode23), it checks whether the current line has a valid header. If the current line has a valid header, the program outputs the XML header and sets the **header** variable to true. Otherwise, the program outputs an error message and exits.

Once the input has encountered a valid header, the program starts parsing each line of code. The program uses **preg_split** to split the line into an array of arguments. Then the program checks the first argument (which is the opcode) and calls the appropriate function to verify the syntax of the remaining arguments. If all the arguments have a valid syntax, the program outputs an XML tag **<instruction>** containing the opcode and the arguments. If any argument has an invalid syntax, the program outputs an error message and exits. The program also keeps track of the order of instructions and outputs an **order** attribute in each **<instruction>**. Finally, the program outputs the closing XML tag **</program>**.