Implementation documentation for IPP 2022/23 university course

Author: Alina Vinogradova
Login: xvinog00

parse.php script

To implement the first part of the project, I used an additional script `scanner.php`. The essence of its work is the line-by-line processing of input data from the `stdin` in `get_token()` function. Each row is subsequently divided into an array, each element of which represents a word in the given string.

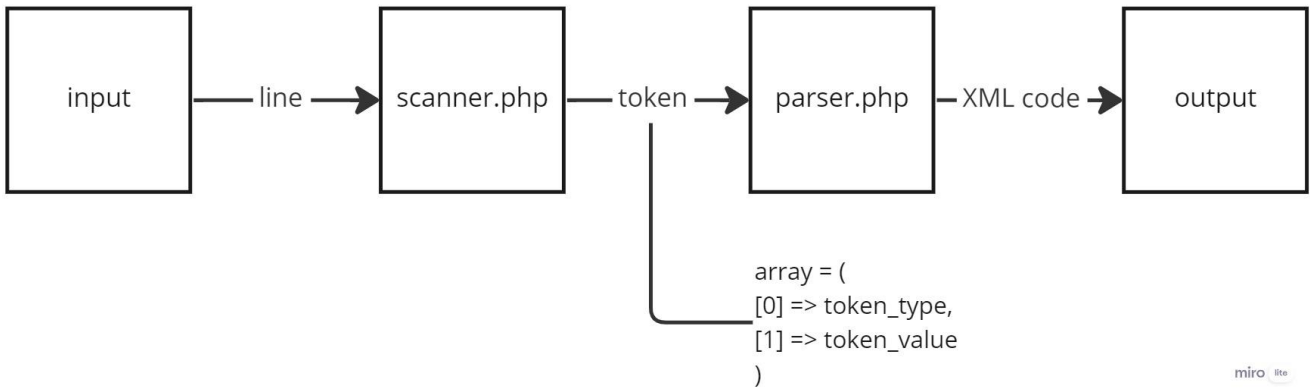

Diagram of the first part of the project

Then, using regular expressions, the scanner transforms each word of a single line into a token, which is represented using an array of two elements: the first is the token type, the second is a specific value (for example, for tokens that are symbols - `tVar` and `tConst` it is value of a given symbol and for tokens of instructions, types and labels - `tOpcode`, `tType` and `tLabel` it is specific data characterizing the object).

Next, the main script cyclically (until the end of the data entry or until one of the errors occurs) calls the `get_token()` function from `scanner.php`. Subsequently, the variable `$token` is processed through the main mechanism of the program, which is in the function `process_token()` - switch statement, which processes the received instructions and checks the correctness of the arguments passed with them (i.e. their number and type) and if the instruction itself belongs to the list of existing `IPPcode23` language instructions. If an error occurs, the program ends instantly and returns a specific code along with a message-description in `stderr` (depending on the error). If the input is processed without errors, the XML code collected during processing is sent to `stdout`.