Department of Computer Science
University of Pretoria

Programming Languages
COS 333

Practical 6: PHP and Perl

October 10, 2023

1 Objectives

This practical aims to achieve the following general learning objective:

- To gain and consolidate some experience writing programs in several different imperative scripting languages, including: PHP and Perl;
- To consolidate a variety of basic concepts related to imperative programming languages and scripting languages, as presented in the prescribed textbook for this course.

2 Plagiarism Policy

Plagiarism is a serious form of academic misconduct. It involves both appropriating someone else's work and passing it off as one's own work afterwards. Thus, you commit plagiarism when you present someone else's written or creative work (words, images, ideas, opinions, discoveries, artwork, music, recordings, computer-generated work, etc.) as your own. Note that using material produced in whole or part by an AI-based tool (such as ChatGPT) also constitutes plagiarism. Only hand in your own original work. Indicate precisely and accurately when you have used information provided by someone else. Referencing must be done in accordance with a recognised system. Indicate whether you have downloaded information from the Internet. For more details, visit the library's website: `http://www.library.up.ac.za/plagiarism/`.

3 Submission Instructions

Upload your practical-related source code files to the appropriate assignment upload slot on the ClickUP course page. For your PHP submission you must implement and submit a single file named `s99999999.php7`, where `99999999` is your student number. For your Perl submission you must implement and submit a single file named `s99999999.pl`, where `99999999` is your student number. Multiple uploads are allowed, but only the last one will be marked. The submission deadline is **Tuesday, 24 October 2023, at 12:00**.

4 Background Information

For this practical, you will be writing programs in PHP 7 (version 7.2.24 is installed in the Informatorium) and Perl 5 (version 26 is installed in the Informatorium). You will have to compare these languages in terms of their support for different programming language concepts. To do this, you will have to write short programs to demonstrate how each language handles the concept under consideration. These programs will not be long, but will demonstrate the concepts adequately, and should allow you to understand the language's support (or lack of support) for the features in question.

Your PHP 7 and Perl 5 programs should run using their respective command line interpreter interfaces on Linux. Be sure to check the version of the interpreter you are using in each case. This is important because in some cases later versions of language specifications are not backward compatible with earlier versions. Manuals (which include tutorials) for PHP [1] and Perl [2] and are available online. The official documentation for PHP (in HTML format) is also provided as additional documentation on the ClickUP page for the course.

5 Practical Tasks

For this practical, you will have to write programs in PHP 7 and Perl 5 to perform the same processing task on the contents of a text file. Each of your scripts must execute using the command line interpreter for the respective scripting language (even though, for example, PHP programs are typically written for execution on a web server). Information on how to use the respective command line interpreters is available via man pages under Linux.

The programs first read an input file (each program should accept a user-provided command line parameter specifying the name of the file to open). The file consists of a series of student numbers and names, separated by a comma and a space. The file has the following general format:

```
1218999, John Stevens
2255559, Jack Timm
2212191, Patricia Mary Daniels
1224312, Joe Black
```

Note that names may consist of any number of first and middle names, but only one last name. A sample input file, named `input.txt`, is provided on the course website (note that your program must work with a similarly formatted input file with an arbitrary number of student details). Your programs must then search the input file and locate the student with the longest last name. In a case where there are ties for the longest last name, the student with the alphabetically earlier last name should be located. Your scripts should then print out only the student number associated with this student. If no longest last name can be found, the output "`None found`" should be produced. There are, of course, several ways in which this search can be performed, and you are free to use whichever approach you prefer. One possible approach to extracting the student number and last name on each line is to use a regular expression.

In the above example, `John Stevens` has a last name that is 7 characters long, `Jack Timm` has a last name that is 4 characters long, `Patricia Mary Daniels` has a last name that is 7 characters long, and `Joe Black` has a last name that is 5 characters long. Therefore, the longest last name is 7 characters long. The program will then select `Patricia Mary Daniels` because `Daniels` is alphabetically before `Stevens`. Both scripts would print out `2212191` (the student number associated with `Patricia Mary Daniels`) as their final results.

Note that you may use any features that are part of the Perl 5 and PHP 7 language installations themselves. **You may not use any third party libraries or extensions**. Also note the following special requirements for the two scripts.

PHP

While PHP programs are typically written for execution on a web server, we will be using the command line interface of PHP 7 to interpret your script. Your script must accept the user input as a command line parameter. The command line parameter is the name of the input file.

When the PHP command line interpreter runs your PHP script, it should generate HTML output which can be opened in a web browser. Note that this output must include HTML tags (it can't just be the text output, even though a Web browser will open a file containing just the output). Also note that this output is generated on the command line, and not output to a file. Check the `php` man page under Linux for information on how to use PHP from the command line.

Perl

We will be using the command line interface of Perl 5 to interpret your script. As for the PHP implementation, your script must accept the user input as a command line parameter, where the parameter is the name of the input file.

Your script should not generate HTML output, as the PHP script does. Instead, only output the required result of the program's execution (either the student number associated with the selected student, or a "`None found`" output).

6 Marking

Each of the implementations will count 5 marks for a total of 10 marks. Submit the PHP and Perl implementations to the appropriate assignment upload slots. Do not upload any additional files other than your source code. Both the implementation and the correct execution of the programs will be taken into account. Your program code will be assessed during the practical session in the week of **Monday, 23 October 2023**.

References

[1] Mehdi Achour, Friedhelm Betz, Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, and Jakub Vrana. PHP manual, 2020. Access at `http://php.net/download-docs.php`.

[2] Perl.org. Perl docs, 2020. Access at `https://www.perl.org/docs.html`.