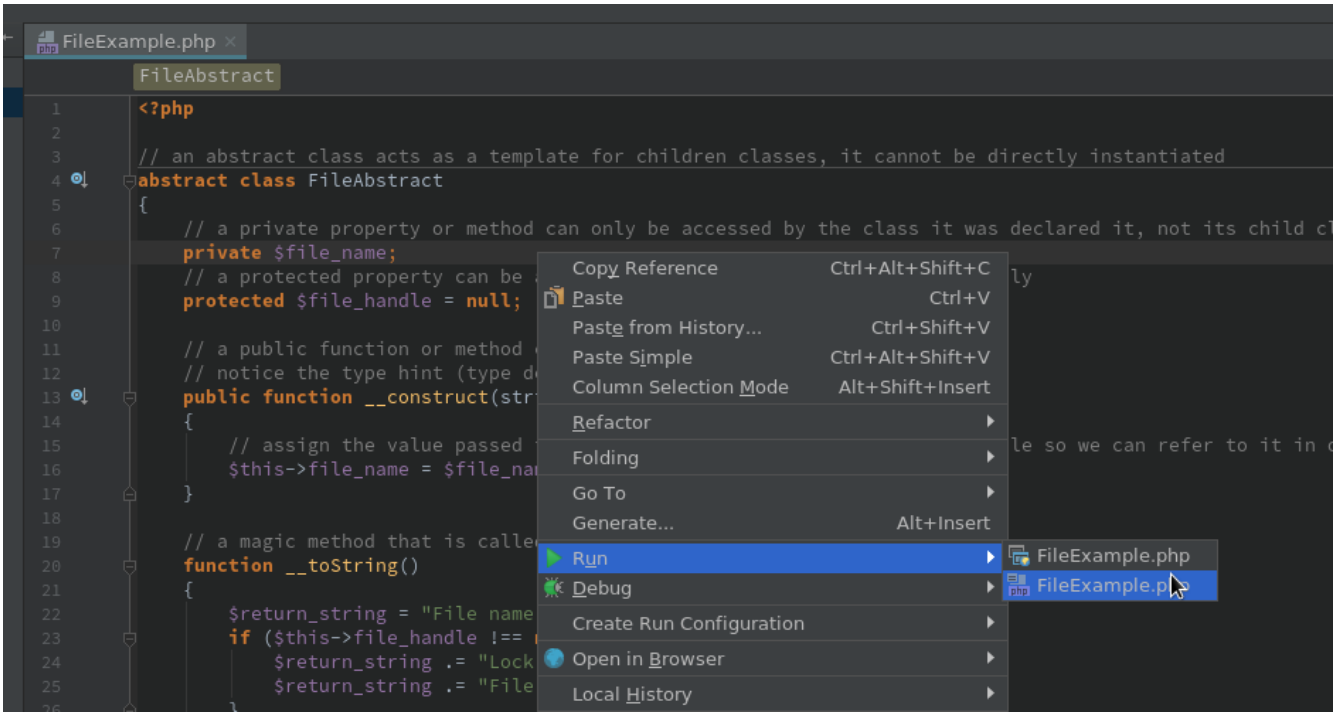
ICS-325 Fall 2017
Assignment 5

For this assignment, will will be running PHP on the command line (CLI). To do so in PHP, right click on your code window, and then select icon with the txt PHP next to it. See the image below:



The example code we went over in class is included in the file FileExample.php. It shows the basics of how to write object oriented PHP with an abstract class and subclasses.

The purpose of the programming assignment is to become comfortable with using the object-oriented features of PHP. You need to create an abstract Report class that 3 other child classes inherit from. Each subclass (child class) will perform an operation on the parameters give to the constructor and will implement various methods from the abstract class to allow the report information to be extracted via method calls from the parent class.

See the accompanied code file: assignment_5_given_code.php

You need to use the Report->printToTerminal() method provided in the code file in your Report class. You should also set the timezone to CST which is also shown in the code file. Finally, all your output should match that shown in the code file. You can copy out parts of the code to test your program. I will run the code file in order to grade your program as well as review your source code.

Each child class needs to implement various abstract methods which are used to generate the report result. Since all the reports should be in the same form, an abstract class Report is used as the parent for all the children.

Abstract Class Report

Properties:

- private $generation_time_stamp

Abstract protected methods:

- getTitle(): should return the title of the report as a string
- getReportDescription(): should return the description of the report as a string
- getParameterString(): should return the parameters given to the report as a string
- getReportResultString(): should return the result of the report computation

Protected methods:

- getGenerationTimeStamp(): returns the value of the private property $generation_time_stamp

Public methods:

- printToTerminal(): prints all the report information to the terminal. Code is given for this method.
- __construct(): set the private instance variable $generation_time_stamp to the current time stamp in the DATE_RFC2822 format (an example is given in the given code).

printToTerminal: should be implemented as (which is provided in the given code file):

```
public function printToTerminal()
{
    echo "Report title: {$this->getTitle()}\n";
    echo "Report generated at: {$this->getGenerationTimeStamp()}\n";
    echo "Report description: {$this->getReportDescription()}\n";
    echo "Report parameters:\n{$this->getParameterString()}\n\n";
    echo "Report result:\n{$this->getReportResultString()}\n";
}
```

Basics of the Subclasses (child classes)

Each child class requires parameters to in order to generate the report. The parameters are passed into the constructor. A private instance variable should be used to store each parameter. The report result can either be generated in the constructor or in the getReportResultString() method.

Class RangeReport extends Report

title: "Number Range Generation"

description: "Prints every integer from START to END separated by a space."

parameter string: "START: $this->start\nEND: $this->end" (make your variable names match)

report result string: The string of integers separated by a command a space, example:
"3, 4, 5"

constructor: first argument is the starting integer, second argument is the ending integer

constructor required validation of the starting and ending integers: Use the filter_var function to ensure that the start and end values are integers and that they are in the range of 0 to 100. You don't need to perform any more validation than that. For example, it is okay if the end value is less than the starting value (that should return an empty string as there is no range value for those parameters).

invalid input action: For both the start and end integers, throw an exception if they fail validation. There should be one exception thrown for when the start value fails with the message: "Bad start value." and another exception thrown for when the end values fails with the message: "Bad end value."

Class SumReport extends Report

title: "Sum Calculator"

description: "Sums the given integers."

parameter string: "INTEGERS: " followed by the list of integers separated by a space and a comma (hint use the array function implode()), example:
"INTEGERS: 2, 4, 6"

report result string: The sum of the integers give in the input array.

constructor: only argument is an array of integers

validation: no validation required

Class MinMaxReport extends Report

title: "Minimum and Maximum Finder"

description: "Find the minimum and maximum of the given integers."

parameter string: "INTEGERS: " followed by the list of integers separated by a space and a comma (hint use the array function implode()), example:
"INTEGERS: 2, 4, 6"

report result: "MINIMUM: $min\nMAXIMUM: $max" (make your variable names match)

constructor: only argument is an array of integers (hint, array sorting can be used to find the min/max)

validation: no validation required

Grading

4 pts Correct definition of the Report class
10 pts Correct definition of the RangeReport class
6 pts Correct definition of the SumReport class
6 pts Correct definition of the MinMaxReport class
6 pts Each report instantiation and printToTerminal call matches the expected output in the given code file. This code is at the bottom of the file, and the expected output is shown in code comments. Your code output should match that could output exactly!

32 pts Total

Note: You may collaborate with other students on this assignment, but you should write and turn in your own code. It is important that you understand the concepts in this assignment in order to do well in future assignments.