**OOP Practical Work 3**

**<?php**

**class CollatzCalculator {**

**public function calculateIterations($x) {**

**$count = 0;**

**while ($x != 1) {**

**if ($x % 2 == 0) {**

**$x /= 2;**

**} else {**

**$x = 3 \* $x + 1;**

**}**

**$count++;**

**}**

**return $count;**

**}**

**}**

**class RangeAnalyzer {**

**private $collatzCalculator;**

**public function \_\_construct() {**

**$this->collatzCalculator = new CollatzCalculator();**

**}**

**public function calculateMaxAndMinValues($start, $end) {**

**$maxValue = 0;**

**$minValue = PHP\_INT\_MAX;**

**$maxNumbers = [];**

**$minNumbers = [];**

**$totalIterations = 0;**

**for ($i = $start; $i <= $end; $i++) {**

**$currentIterations = $this->collatzCalculator->calculateIterations($i);**

**$totalIterations += $currentIterations;**

**if ($currentIterations > $maxValue) {**

**$maxValue = $currentIterations;**

**$maxNumbers = [$i];**

**} elseif ($currentIterations == $maxValue) {**

**$maxNumbers[] = $i;**

**}**

**if ($currentIterations < $minValue) {**

**$minValue = $currentIterations;**

**$minNumbers = [$i];**

**} elseif ($currentIterations == $minValue) {**

**$minNumbers[] = $i;**

**}**

**}**

**return [**

**'max\_value' => $maxValue,**

**'max\_numbers' => $maxNumbers,**

**'min\_value' => $minValue,**

**'min\_numbers' => $minNumbers,**

**'total\_iterations' => $totalIterations**

**];**

**}**

**protected function mathematicProgression($start, $end, $step) {**

**$sequence = [];**

**for ($i = $start; $i <= $end; $i += $step) {**

**$sequence[] = $i;**

**}**

**return $sequence;**

**}**

**}**

**class ExtendedRangeAnalyzer extends RangeAnalyzer {**

**public function calculateStatistics($start, $end) {**

**$data = $this->calculateMaxAndMinValues($start, $end);**

**$averageIterations = $data['total\_iterations'] / ($end - $start + 1);**

**return [**

**'max\_value' => $data['max\_value'],**

**'max\_numbers' => $data['max\_numbers'],**

**'min\_value' => $data['min\_value'],**

**'min\_numbers' => $data['min\_numbers'],**

**'total\_iterations' => $data['total\_iterations'],**

**'average\_iterations' => $averageIterations**

**];**

**}**

**}**

**// Example usage:**

**$extendedAnalyzer = new ExtendedRangeAnalyzer();**

**$statistics = $extendedAnalyzer->calculateStatistics(1, 100);**

**print\_r($statistics);**