# Zadanie rekrutacyjne - staż

Zadanie polega na implementacji klas:

- `Series` obsługującej statystyki ocen poszczególnych odcinków serialu telewizyjnego.
- `Episode` przechowującej dane pojedynczego odcinka.
- `CsvAdapter` ładującej statystyki zapisane w formacie CSV.

Wycinek pliku `ratings.csv`[1] zawierającego dane:

```
s,e,10,9,8,7,6,5,4,3,2,1,title
1,00,1664,1054,1484,832,344,141,56,38,63,54,100Pilot
1,01,425,423,617,300,102,26,11,5,0,4,101DeepThro
1,10,334,298,520,282,94,25,12,5,2,11,110Eve
```

Znaczenie kolumn opisuje poniższa tabela.

| **Nazwa kolumny** | **Opis kolumny** |
|---|---|
| s | Numer kolejny sezonu |
| e | Numer kolejny odcinka |
| 10...1 | Liczba głosów na daną ocenę |
| title | Skrócony tytuł odcinka |

[1] Źródło: https://github.com/zimolzak/xfiles

Klasa `Series` musi implementować interfejs `SeriesInterface`:

```
interface SeriesInterface
{
      public function __construct(Iterator $episodes);

      /**
       * Returns best rated episode for specified $season number.
       *
       * @param int $season
       * @return EpisodeInterface
       */
      public function getBestInSeason($season);

      /**
       * Returns best rated season finale episode.
       *
       * @return EpisodeInterface
       */
      public function getBestSeasonFinale();

      /**
       * Returns array of episodes starting with $letter.
       *
       * @param string $letter
       * @return array of EpisodeInterface
       */
      public function getByTitleFirstLetter($letter);
}
```

Klasa `Episode` musi implementować interfejs `EpisodeInterface`:

```
interface EpisodeInterface
{
	/**
	 * @param int $season Season number.
	 * @param int $number Episode number.
	 * @param float $rating Episode rating.
	 * @param string $title Episode title.
	 */
	public function __construct($season, $number, $rating, $title);

	/**
	 * Returns season number.
	 *
	 * @return int
	 */
	public function getSeason();

	/**
	 * Returns episode number.
	 *
	 * @return int
	 */
	public function getNumber();

	/**
	 * Returns weighted arithmetic mean of episode ratings.
	 *
	 * @return float
	 */
	public function getRating();

	/**
	 * Returns episode title.
	 *
	 * @return string
	 */
	public function getTitle();
}
```

Klasa `CsvAdapter` musi implementować interfejs `Iterator`.

Po umieszczeniu plików klas w tym samym folderze i uruchomieniu pliku `index.php` zawarty w nim kod powinien wykonać się bezbłędnie oraz wyświetlić wynik przedstawiony na następnej stronie.

```
<?php

require_once 'CsvAdapter.php';
require_once 'EpisodeInterface.php';
require_once 'Episode.php';
require_once 'SeriesInterface.php';
require_once 'Series.php';

$series = new Series(new CsvAdapter('ratings.csv'));

$p = $series->getByTitleFirstLetter('p');

?>
<table border=”1”>
     <tbody>
          <tr>
               <th>Best in s05</th>
               <td><?= $series->getBestInSeason(5) ?></td>
          </tr>
          <tr>
               <th>Best season finale</th>
               <td><?= $series->getBestSeasonFinale() ?></td>
          </tr>
          <tr>
               <th>Starting with P</th>
               <td>
                    <?php if ( ! empty($p)): ?>
                    <ul>
                         <?php foreach ($p as $title): ?>
                         <li><?= $title ?></li>
                         <?php endforeach; ?>
                    </ul>
                    <?php else: ?>
                    Nothing found
                    <?php endif; ?>
               </td>
          </tr>
     </tbody>
</table>
```

| | |
|---|---|
| **Best in s05** | 512BadBlood |
| **Best season finale** | 225Anasazi |
| **Starting with P** | ● 100Pilot<br>● 315PiperMar<br>● 317Pusher<br>● 302PaperCli<br>● 410PaperHea<br>● 513PatientX<br>● 813PerManum<br>● 803Patience<br>● 910Providen<br>● 909Provenan |

Tam gdzie jest to istotne o pozycji w rankingu decyduje średnia ważona ocen obliczona dla danego odcinka.

# Powodzenia!