**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ**

**КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ №1**

з дисципліни:« Проектування веб-орієнтованих інформаційних систем**»**

Виконав: студент групи ЦТ.м-51н

Кириченко Д.С.

Варіант: 8

Перевірила: Парфененко Ю.В.

Суми 2025

ТЕМА: Створення класів та об’єктів мовою PHP

Варіант 8 - Робітник (вік, прізвище, підприємство).

Завдання 3. Створіть клас з полями згідно з Вашим варіантом. Для полів задати специфікатори доступу public. Реалізувати в  класі методи:

⦁ set()  –  встановлює  значення поля класу;

⦁ get() – повертає значення поля класу;

⦁ show()– виводить на екран значення поля класу;

⦁ search() – здійснює пошук за одним із полів класу.

Продемонструйте роботу з класом, створивши 3 об’єкти. Задайте полям класу довільні значення.

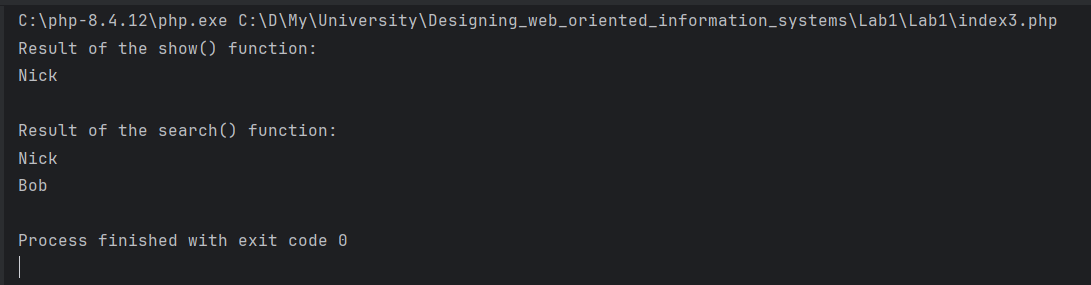


Рисунок 1 – Результат роботи функцій show() та search()

⦁ Додайте у клас ще один приватний метод та продемонструйте механізм звернення до цього методу усередині класу.

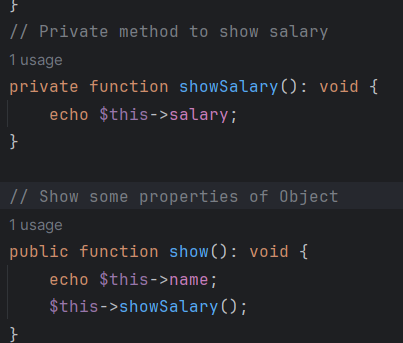


Рисунок 2 – Приватний метод showSalary() та звернення до нього

⦁ Додайте у клас ще одне поле та задайте для нього специфікатор доступу private. Продемонструйте з використанням цього поля механізм інкапсуляції.

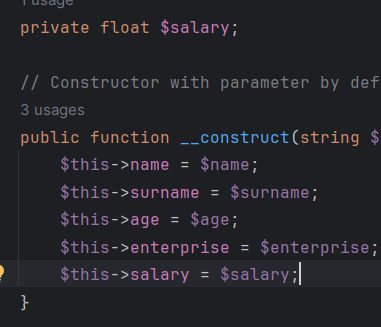


Рисунок 3 – Звернення до приватного поля salary у конструкторі класу

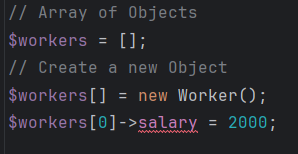


Рисунок 4 – Звернення до приватного поля salary

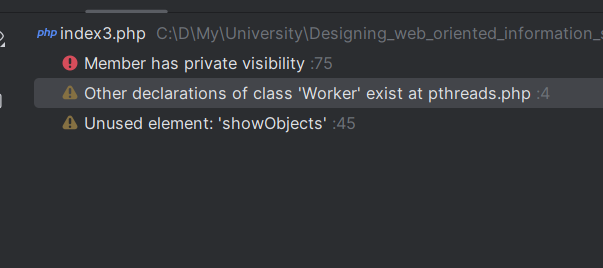


Рисунок 5 – Помилка при зверненні, якщо поле має специфікатор доступу private

⦁ Створіть масив, що складається з 5 об’єктів класу. Реалізувати метод show\_objects(), який виводить масив об’єктів класу на екран.

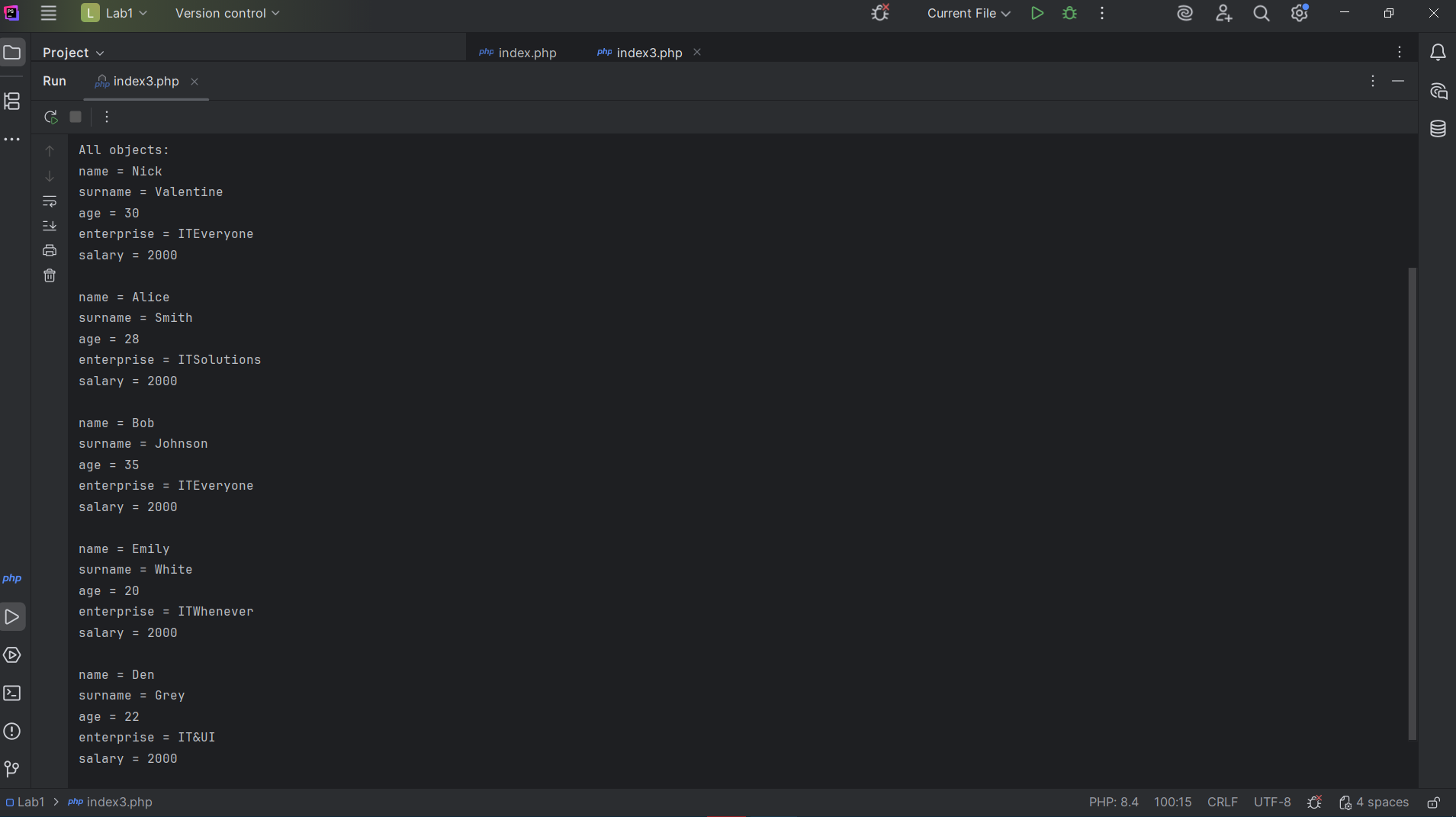


Рисунок 5 – Результат роботи функції showObjects()

Програмний код:

<?php  
  
// Variant 8  
class Worker  
{  
 // Four public properties: name, surname, age, enterprise and one private - salary  
 public string $name;  
 public string $surname;  
 public int $age;  
 public string $enterprise;  
 private float $salary;  
  
 // Constructor with parameter by default for each property  
 public function \_\_construct(string $name = "Nick", string $surname = "Valentine", int $age = 30, string $enterprise = "ITEveryone", float $salary = 2000) {  
 $this->name = $name;  
 $this->surname = $surname;  
 $this->age = $age;  
 $this->enterprise = $enterprise;  
 $this->salary = $salary;  
 }  
  
 // Universal get function for every property  
 public function \_\_get(string $property): mixed {  
 // Check whether the property exists  
 if(property\_exists($this, $property)) {  
 return $this->$property;  
 }  
 return null;  
 }  
  
 // Universal setter for each property  
 public function \_\_set(string $property, mixed $value): void {  
 // Check whether the property exists  
 if(property\_exists($this, $property)) {  
 $this->$property = $value;  
 }  
 }  
 // Private method to show salary  
 private function showSalary(): void {  
 echo $this->salary;  
 }  
  
 // Show some properties of Object  
 public function show(): void {  
 echo $this->name;  
 $this->showSalary();  
 }  
  
 // Show print every property of Object  
 public function showObjects(): void {  
 // get every property from current Object  
 $properties = get\_object\_vars($this);  
 // print every property, for example: name = Nick  
 foreach ($properties as $property => $value) {  
 echo "$property = $value\n";  
 }  
 }  
  
 // Search for any Objects by any property  
 public static function search(array $workers, string $property, mixed $value): array {  
 // Array for saving result  
 $result = [];  
 // Go through each Object in the array  
 foreach ($workers as $worker) {  
 // Check whether a value of specified property equals to a specified value  
 if($worker->$property === $value) {  
 // Add Object to the end of the array  
 $result[] = $worker;  
 }  
 }  
  
 return $result;  
 }  
}  
  
// Array of Objects  
$workers = [];  
// Create a new Object  
$workers[] = new Worker();  
$workers[] = new Worker("Alice", "Smith", 28, "ITSolutions");  
$workers[] = new Worker("Bob", "Johnson", 35, "ITEveryone");  
$workers[] = new Worker("Emily", "White", 20, "ITWhenever");  
$workers[] = new Worker("Den", "Grey", 22, "IT&UI");  
  
// Show every property of an Object worker  
echo "Result of the show() function:\n";  
$workers[0]->show();  
// Find every Object with by specified property "enterprise" with value "ITEveryone"  
$searchResult = Worker::*search*($workers, "enterprise", "ITEveryone");  
  
// Show the search results  
echo "\n\nResult of the search() function:\n";  
foreach ($searchResult as $worker) {  
 echo $worker->show() . "\n";  
}  
  
// Show every property for every Object  
echo "\n\nAll objects:\n";  
foreach ($workers as $worker) {  
 $worker->showObjects();  
 echo "\n";  
}

4. Створіть у файлі class.php користувальницький клас Coor із полем даних $name, конструктором та методом Getname. Створити об’єкт цього класу, надати йому значення «Nick» та вивести це значення на екран.

Внесіть зміни до попереднього файла, застосувавши функцію unset для знищення об’єкта класу. У разі успішного знищення виведіть на екран напис «Object is deleted!». Змінений файл збережіть.

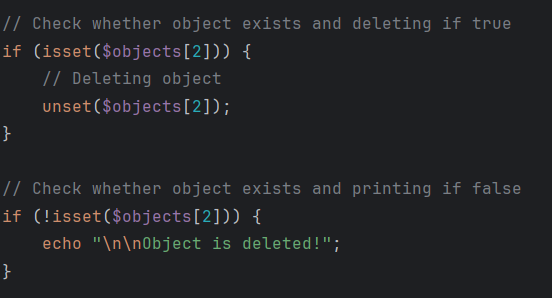


Рисунок 6 – Видалення об’єкта та перевірка його існування

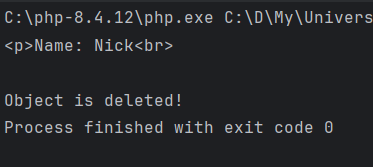


Рисунок 7 – Результат роботи unset()

Внесіть зміни до попереднього файла, застосувавши замість функції unset деструктор, який буде друкувати повідомлення, що об'єкт класу знищено.

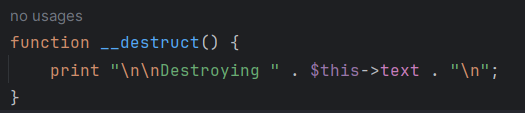


Рисунок 8 - Деструктор

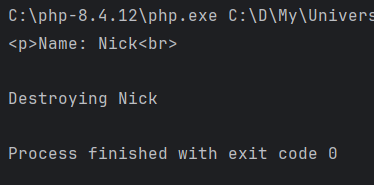


Рисунок 9 – Результат роботи деструктора

Додайте у клас Coor ще два поля даних – $login та $password. Створіть три об’єкти класу та виведіть їх на екран.

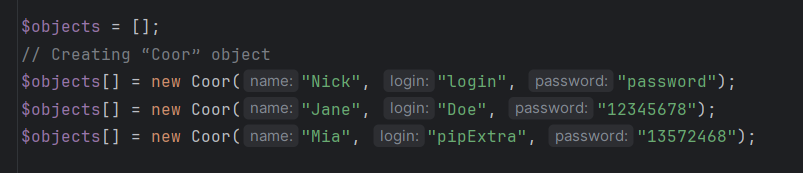


Рисунок 10 – Створення нових об’єктів

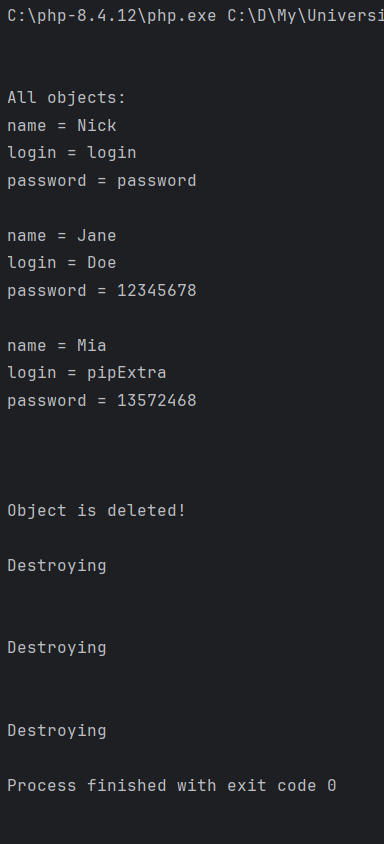


Рисунок 11 - Результат роботи програми

Програмний код:

<?php

class Coor  
{  
 private string $name;  
 private string $login;  
 private string $password;  
  
 // Function for initialize properties  
 function \_\_construct($name, $login, $password) {  
 $this->name=$name; //set some “name” to this “name”;  
 $this->login=$login; //set some “login” to this “login”;  
 $this->password=$password; //set some “password” to this “password”;  
 }  
  
 // Function for getting properties  
 function \_\_get($property): mixed {  
 if (property\_exists($this, $property)) {  
 return $this->$property;  
 }  
 return null;  
 }  
  
 // Show print every property of Object  
 public function showObjects(): void {  
 // get every property from current Object  
 $properties = get\_object\_vars($this);  
 // print every property, for example: name = Nick  
 foreach ($properties as $property => $value) {  
 echo "$property = $value\n";  
 }  
 }  
  
 // Class destructor  
 function \_\_destruct() {  
 print "\n\nDestroying " . $this->text . "\n";  
 }  
  
}  
  
$objects = [];  
// Creating “Coor” object  
$objects[] = new Coor("Nick", "login", "password");  
$objects[] = new Coor("Jane", "Doe", "12345678");  
$objects[] = new Coor("Mia", "pipExtra", "13572468");  
  
// Getter call  
$objects[0]->text;  
  
// Show every property for every Object  
echo "\n\nAll objects:\n";  
foreach ($objects as $object) {  
 $object->showObjects();  
 echo "\n";  
}  
  
  
// Check whether object exists and deleting if true  
if (isset($objects[2])) {  
 // Deleting object  
 unset($objects[2]);  
}  
  
// Check whether object exists and printing if false  
if (!isset($objects[2])) {  
 echo "\n\nObject is deleted!";  
}

5. Створіть текстовий файл count.txt з довільним вмістом. Створіть клас для роботи з текстовими файлами. Реалізуйте конструктор, метод, що виводить вміст файлу на екран, метод, що виводить розмір файлу та метод, що виводить кількість рядків у файлі.

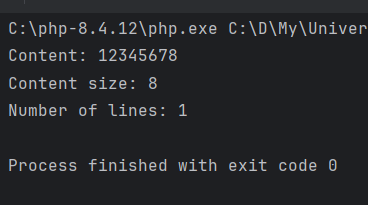


Рисунок 12 – Виведення основних даних файлу на екран

6. Створіть клас для  роботи з текстовим файлом. Реалізувати у класі конструктор та метод виконання завдання згідно з Вашим варіантом.

Варіант 8. Дано текстовий файл, що складається з декількох рядків. Додати новий рядок у кінець файлу.

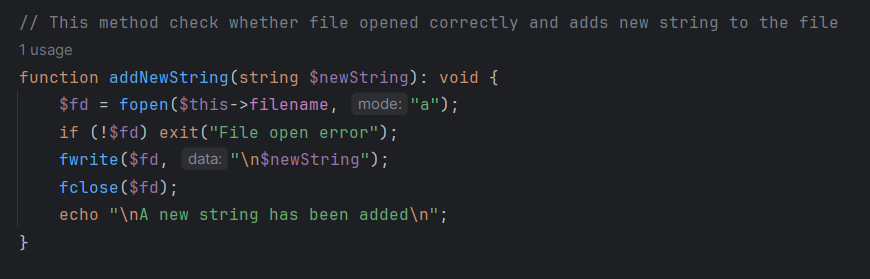


Рисунок 13 – Додавання нового рядка до файлу

Програмний код:

<?php  
  
header("Content-Type: text/html; charset=windows-1251");  
  
class WorkWithFile  
{  
 // Two public properties for file content and file directory  
 public string $buff;  
 public string $filename;  
  
 // Constructor to open file and check whether it opened correctly  
 function \_\_construct($filename) {  
 $uploadDir = './';  
 $this->filename = $uploadDir . $filename;  
 if (!file\_exists($this->filename)) exit("File does not exist");  
  
 // File opening  
 $fd = fopen($filename, "r");  
  
 if (!$fd) exit("File open error");  
  
 $this->buff = fread($fd, filesize($this->filename));  
 fclose($fd);  
 }  
  
 // The method displays the contents of the //file on the function screen  
 function getContent(): string {  
 return $this->buff;  
 }  
  
 // The method displays the file size  
 function getSize(): int {  
 return filesize($this->filename);  
 }  
  
 // The method outputs the number of lines in the //function file  
 function getCount(): int {  
 if (!empty($this->filename)) {  
 $arr = file($this->filename);  
 return count($arr);  
 }  
 else  
 return 0;  
 }  
  
 // This method check whether file opened correctly and adds new string to the file  
 function addNewString(string $newString): void {  
 $fd = fopen($this->filename, "a");  
 if (!$fd) exit("File open error");  
 fwrite($fd, "\n$newString");  
 fclose($fd);  
 echo "\nA new string has been added\n";  
 }  
}  
  
// Create new object  
$first = new WorkWithFile("count.txt");  
  
// Print object properties  
echo "Content: {$first->getContent()}\n";  
echo "Content size: {$first->getSize()}\n";  
echo "Number of lines: {$first->getCount()}\n";  
  
// Add new string to the file  
$first->addNewString("87654321");

7. Створіть у Excel таблицю, з полями, зображеними на рис.1.

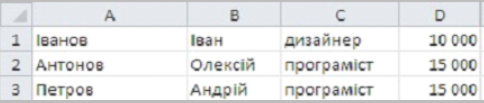


Рисунок 14 – Таблиця з даними

Збережіть таблицю як файл file.csv з роздільником поля «;».

Створіть клас для роботи з файлами формату .csv. Реалізуйте конструктор, метод, що виводить вміст файлу на екран та метод, що додає інформацію в файл.

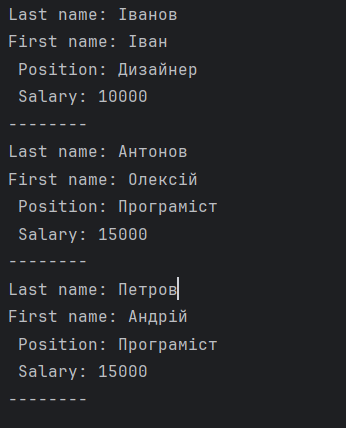


Рисунок 15 – Вивід вмісту csv файлу на екран

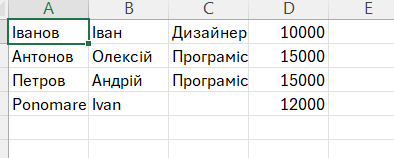


Рисунок 16 – Додавання вмісту до csv файлу

Програмний код:

<?php  
  
class CSV  
{  
 // Private property for csv file  
 private $\_csv\_file = null;  
 // DOC comments  
 */\*\*  
  
 \* @param string $csv\_file - path to csv file  
  
 \*/* // Constructor to open file correctly  
 public function \_\_construct($csv\_file) {  
 if (file\_exists($csv\_file)) {  
 $this->\_csv\_file = $csv\_file;  
 }  
 else  
 throw new Exception("File not found");  
 }  
  
  
 // Writing data to csv file from array  
 public function setCSV(Array $csv) {  
 $handle = fopen($this->\_csv\_file, "a"); // Open the file for writing  
 foreach ($csv as $value) {  
 fputcsv($handle, explode(";", $value), ";");  
 }  
  
 fclose($handle);  
 }  
  
 // Reading csv file to array  
 public function getCSV() {  
 $handle = fopen($this->\_csv\_file, "r"); // Open the file for reading  
 $array\_line\_full = array();  
 while (($line = fgetcsv($handle, 0, ";")) !== FALSE) {  
 $array\_line\_full[] = $line;  
 }  
  
 fclose($handle);  
 return $array\_line\_full;  
 }  
  
}  
  
// Construction try catch for exceptions  
try {  
 // Create new object  
 $csv = new CSV("file.csv");  
 // Read csv file to array  
 $get\_csv = $csv->getCSV();  
  
 // Print data from array  
 foreach ($get\_csv as $value) {  
 echo "Last name: " . $value[0] . "\n";  
 echo "First name: " . $value[1] . "\n";  
 echo " Position: " . $value[2] . "\n";  
 echo " Salary: " . $value[3] . "\n";  
 echo "--------\n";  
  
 }  
  
 // The array that will be written to csv file  
 $arr = array("Ponomarenko;Ivan;;12000",);  
 $csv->setCSV($arr);  
}  
  
// Catch exception and print exception info  
catch (Exception $e) {  
 echo "Помилка: " . $e->getMessage();  
}

8. Організуйте зберігання даних з предметної області завдання 3 лабораторної роботи у файлі .csv (щонайменше 3 записи). Створість клас для роботи з цим файлом та продемонструйте його роботу.



Рисунок 17 – Дані для запису у csv файл та цикл запису у сам файл

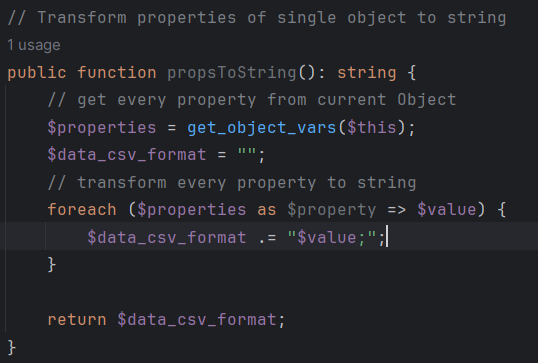


Рисунок 18 – Функція перетворення властивостей об’єкту у рядок з роздільником

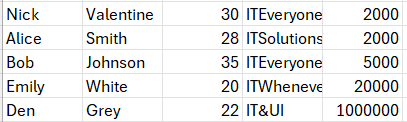


Рисунок 19 – Результат роботи програми. Запис у csv файл

Програмний код:

Файл Worker.php

<?php  
  
class Worker  
{  
 // Four public properties: name, surname, age, enterprise and one private - salary  
 public string $name;  
 public string $surname;  
 public int $age;  
 public string $enterprise;  
 private float $salary;  
  
 // Constructor with parameter by default for each property  
 public function \_\_construct(string $name = "Nick", string $surname = "Valentine", int $age = 30, string $enterprise = "ITEveryone", float $salary = 2000) {  
 $this->name = $name;  
 $this->surname = $surname;  
 $this->age = $age;  
 $this->enterprise = $enterprise;  
 $this->salary = $salary;  
 }  
  
 // Universal get function for every property  
 public function \_\_get(string $property): mixed {  
 // Check whether the property exists  
 if(property\_exists($this, $property)) {  
 return $this->$property;  
 }  
 return null;  
 }  
  
 // Universal setter for each property  
 public function \_\_set(string $property, mixed $value): void {  
 // Check whether the property exists  
 if(property\_exists($this, $property)) {  
 $this->$property = $value;  
 }  
 }  
 // Private method to show salary  
 private function showSalary(): void {  
 echo $this->salary;  
 }  
  
 // Show some properties of Object  
 public function show(): void {  
 echo $this->name;  
 $this->showSalary();  
 }  
  
 // Transform properties of single object to string  
 public function propsToString(): string {  
 // get every property from current Object  
 $properties = get\_object\_vars($this);  
 $data\_csv\_format = "";  
 // transform every property to string  
 foreach ($properties as $property => $value) {  
 $data\_csv\_format .= "$value;";  
 }  
  
 return $data\_csv\_format;  
 }  
  
 // Search for any Objects by any property  
 public static function search(array $workers, string $property, mixed $value): array {  
 // Array for saving result  
 $result = [];  
 // Go through each Object in the array  
 foreach ($workers as $worker) {  
 // Check whether a value of specified property equals to a specified value  
 if($worker->$property === $value) {  
 // Add Object to the end of the array  
 $result[] = $worker;  
 }  
 }  
  
 return $result;  
 }  
}

Файл index8.php

<?php  
  
// Class Worker  
include "Worker.php";  
  
class CSV  
{  
 // Private property for csv file  
 private $\_csv\_file = null;  
 // DOC comments  
 */\*\*  
 \* @param string $csv\_file - path to csv file  
 \*/* // Constructor to open file correctly  
 public function \_\_construct($csv\_file)  
 {  
 if (file\_exists($csv\_file)) {  
 $this->\_csv\_file = $csv\_file;  
 } else  
 throw new Exception("File not found");  
 }  
  
 // Writing data to csv file from array  
 public function setCSV(array $csv)  
 {  
 $handle = fopen($this->\_csv\_file, "a"); // Open the file for writing  
 foreach ($csv as $value) {  
 fputcsv($handle, explode(";", $value), ";");  
 }  
  
 fclose($handle);  
 }  
  
 // Reading csv file to array  
 public function getCSV()  
 {  
 $handle = fopen($this->\_csv\_file, "r"); // Open the file for reading  
 $array\_line\_full = array();  
 while (($line = fgetcsv($handle, 0, ";")) !== FALSE) {  
 $array\_line\_full[] = $line;  
 }  
  
 fclose($handle);  
 return $array\_line\_full;  
 }  
  
}  
  
// Construction try catch for exceptions  
try {  
 // Create new object  
 $csv = new CSV("file1.csv");  
  
 // Array of objects  
 $workers = [];  
 // Create a new Object  
 $workers[] = new Worker();  
 $workers[] = new Worker("Alice", "Smith", 28, "ITSolutions", 2000);  
 $workers[] = new Worker("Bob", "Johnson", 35, "ITEveryone", 5000);  
 $workers[] = new Worker("Emily", "White", 20, "ITWhenever", 20000);  
 $workers[] = new Worker("Den", "Grey", 22, "IT&UI", 1000000);  
  
 // Write every property of object to csv file  
 foreach ($workers as $worker) {  
 $csv->setCSV(array($worker->propsToString()));  
 }  
  
} // Catch exception and print exception info  
catch (Exception $e) {  
 echo "Помилка: " . $e->getMessage();  
}