|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***ЛАБОРАТОРНА РОБОТА №4***  **Об'єктно-орієнтоване програмування в РНР**  ***Мета роботи:*** Навчитися працювати з класами в мові прграмування PHP  **Завдання до виконання:**    ***Лістинг програми:***  Controllers  <body>      <?php          class Fruit {              public $name;              function set\_name($name) {                  $this->name = $name;              }              function get\_name() {                  return $this->name;              }          }          $apple = new Fruit();          $apple->set\_name('Apple');          echo "Value from object -> " . $apple->get\_name();      ?>  </body>  Models  <body>      <?php          class Color {              public $name = "Yellow";              function output\_name() {                  echo("output name: " . $this->name);              }          }          $color = new Color();          $color->output\_name();      ?>  </body>  Views  <body>      <?php          class Animal {              public $name;              function output\_name() {                  echo("Animal name: " . $this->name);              }          }          $animal = new Animal();          $animal->name = "Wolf";          $animal->output\_name();      ?>  </body>    **Клас Human:**  **body**  <?php  namespace Human;  class Body{      public $heart;      public $stomach;      function heart\_beat(){          echo "Hear beating" . "<br>";      }  }  **head**  <?php  namespace Human;  class Head{      public $brain = "think";      public $eyes = "look";      function look\_at(){          echo "Taking " . $this->eyes . " at...";      }      function think(){          echo " car and " . $this->brain . "ing about having it.";      }  }  **Реалізація автопідключення класів:**  <?php      function myAutoloader($className) {          $classFile = 'classes/'. $className . '.php';            if (file\_exists($classFile))              require\_once($classFile);      }        spl\_autoload\_register('myAutoloader');        $a = New \Human\Body();      $b = New \Human\Head();      $a->heart\_beat();      $b->look\_at();      $b->think();    ***Лістинг програми:***  <?php  class Circle{      private $c;      private $r;      private $x;      private $y;      function \_\_construct($x, $y, $r){          $this->x = $x;          $this->y = $y;          $this->r = $r;      }      function \_\_toString(){          return "Коло з центром в {$this->x}, {$this->y} і радіусом {$this->r}";      }      function getX(){          return $this->x;      }      function setX($x){          $this->x = $x;      }      function getY(){          return $this->y;      }      function setY($y){          $this->y = $y;      }      function getR(){          return $this->r;      }      function setR($r){          $this->r = $r;      }      function intersect($circle){          $distance\_between\_centers = sqrt(pow($this->x - $circle->x, 2) + pow($this->y - $circle->y, 2));            if ($distance\_between\_centers > $this->r + $circle->r) {              return false;          } else {              return true;          }      }  }  $circle1 = New Circle(20, 0.20, 15);  $circle2 = new Circle(6, 0, 3);  echo $circle1->\_\_toString() . "<br>";  echo $circle1->getR() . "<br>";  echo $circle1->getX() . "<br>";  echo $circle1->getY() . "<br>";  echo $circle1->intersect($circle2) ? 'Перетинаються' : 'Не перетинаються';  Рисунок 1.1 – Результат виконання програми      ***Лістинг програми:***  <?php  class MyClass {      public static $dir = "text/";      static function read($name){          $content = file\_get\_contents(self::$dir . $name . ".txt");          echo $content;      }      static function write($name, $text){          $file = fopen(self::$dir . $name . ".txt", 'a');          fwrite($file, $text . "\n");          fclose($file);      }      static function clear($name){          file\_put\_contents(self::$dir . $name . ".txt", "");      }  }  MyClass::write("1", "hello");  MyClass::write("1", "Hi");  MyClass::read("1");  MyClass::clear("1");  Рисунок 1.2 – Результат виконання програми      ***Лістинг програми:***  <?php  abstract class Human{      private $name;      private $height;      private $weight;      private $age;        protected abstract function birthMessage();      function setName($name){          $this->name = $name;      }      function getName(){          return $this->name;      }      function setHeight($height){          $this->height = $height;      }      function getHeight(){          return $this->height;      }      function setWeight($weight){          $this->weight = $weight;      }      function getWeight(){          return $this->weight;      }      function setAge($age){          $this->age = $age;      }      function getAge(){          return $this->age;      }      public function giveBirth() {          $this->birthMessage();      }  }  class Student extends Human{      private $HEI;      private $course = 0;      function \_\_construct($name, $age, $height, $weight){          $this->setName($name);          $this->setAge($age);          $this->setWeight($weight);          $this->setHeight($height);      }      function setHEI($HEI){          $this->HEI = $HEI;      }      function getHEI(){          return $this->HEI;      }      function setCourse($course){          $this->course = $course;      }      function getCourse(){          return $this->course;      }      function updCourse(){          if($this->course < 4){              $this->course ++;          }      }      protected function birthMessage() {          echo "Вітаємо, у вас народилась дитина!";      }  }  class Programmer extends Human{      private $programmer\_lang;      private $Experience;      function \_\_construct(){          $this->programmer\_lang = array();          $this->programmer\_lang[] = "C++";          $this->programmer\_lang[] = "Python";          $this->Experience = 2;      }      function setExperience($Experience){          $this->Experience = $Experience;      }      function getExperience(){          return $this->Experience;      }      function getProgrammerlang(){          return $this->programmer\_lang;      }      function addLanguage($programmer\_lang, $element){          $this->programmer\_lang[] = $element;      }      protected function birthMessage() {          echo "Вітаємо, у вас народилась дитина!";      }  }  $student = new Student("John", 20, 180, 75);  $student->setHEI("Example University");  $student->setCourse(2);  echo "Student Info:<br>";  echo "Name: " . $student->getName() . "<br>";  echo "Age: " . $student->getAge() . "<br>";  echo "Height: " . $student->getHeight() . " cm<br>";  echo "Weight: " . $student->getWeight() . " kg<br>";  echo "HEI: " . $student->getHEI() . "<br>";  echo "Course: " . $student->getCourse() . "<br>";  echo "<br>" . $student->giveBirth() . "<br>";  // Оновлюємо курс студента  $student->updCourse();  echo "Updated Course: " . $student->getCourse() . "<br>";  echo "<hr>";  // Створюємо програміста  $programmer = new Programmer();  $programmer->setName("Alice");  $programmer->setAge(25);  $programmer->setHeight(170);  $programmer->setWeight(65);  echo "Programmer Info:<br>";  echo "Name: " . $programmer->getName() . "<br>";  echo "Age: " . $programmer->getAge() . "<br>";  echo "Height: " . $programmer->getHeight() . " cm<br>";  echo "Weight: " . $programmer->getWeight() . " kg<br>";  echo "Experience: " . $programmer->getExperience() . " years<br>";  echo "<br>" . $programmer->giveBirth() . "<br>";  // Додаємо мову програмування  $programmer->addLanguage("Java", "PHP");  echo "Programmer Languages:<br>";  $languages = $programmer->getProgrammerlang();  foreach ($languages as $language) {      echo $language . "<br>";  }  Рисунок 1.3 – Результат виконання програми      ***Лістинг програми:***  <?php  interface HouseCleaning {      public function cleanRoom();      public function cleanKitchen();  }  abstract class Human implements HouseCleaning {      function \_\_construct(){          echo "Human class created <br><br>";      }  }  class Student extends Human {      public function cleanRoom() {          echo "Студент прибирає кімнату<br>";      }      public function cleanKitchen() {          echo "Студент прибирає кухню<br>";      }  }  class Programmer extends Human {      public function cleanRoom() {          echo "Програміст прибирає кімнату<br>";      }      public function cleanKitchen() {          echo "Програміст прибирає кухню<br>";      }  }  $student = new Student();  $student->cleanRoom();  $student->cleanKitchen();  $programmer = new Programmer();  $programmer->cleanRoom();  $programmer->cleanKitchen();  Рисунок 1.4 – Результат виконання програми | | | | | | | | | | |
|  |  |  |  |  | *ДУ «Житомирська політехніка».21.121.13.00 – Лр4* | | | | | |
|  |  |  |  |  |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |
| *Розроб.* | | *Павленко Н.В.* |  |  | Звіт з лабораторної роботи | *Літ.* | | | *Арк.* | *Аркушів* |
| *Перевір.* | | *Дмитренко І.А.* |  |  |  |  |  |  |  |
| *Керівник* | |  |  |  | *ФІКТ Гр. IПЗк-23-1* | | | | |
| *Н. контр.* | |  |  |  |
| *Зав. каф.* | |  |  |  |