Challenge - PHP OOP

For this week's challenge we need to write code for managing furniture. Please consider the following general guidelines when writing your code:

- Do not duplicate your code.
- Try to use the parent methods whenever possible.

Part 1: Create a class named Furniture.

- This class should have 5 private properties:
  width, length, height, is_for_seating, is_for_sleeping.
- The values for width, height and length properties should be set through the class constructor.
- The values for is_for_seat and is_for_sleep parameters should be set using the appropriate getter and setter methods.
- Create two methods: *area*() and *volume*().
- The *area* method should returns the *width * length* value and the *volume* method should return the *area * height* value.
- Instantiate an object from the class Furniture and call all the methods.

Part 2: Create a class named Sofa that extends the class Furniture.

- This class has 3 additional properties: seats, armrests, length_opened.
- The constructor accepts the same arguments as the parent's constructor.
- For the additional properties, write setter and getter methods.
- Create one method called *area_opened()*
  - ➢ This method should check if the sofa is suitable for sleeping.
  - ➢ If yes, the method should return the width * length_opened value.
  - ➢ If not, it should return:
    'This sofa is for sitting only, it has X armrests and X seats'.
- Create an object from the class Sofa and set the sofa as sitting only.

- Then call the methods area(), volume() and area_opened()
- Change the sofa to be appropriate for sleeping (is_for_sleeping = 1) and set its length when opened through the property's setter method.
- Call the area_opened() method again.

Part 3: Create Bench and Chair classes.

- The Bench class should extend the Sofa class.
- The Chair class should extend the Furniture class.
- Both classes should have a constructor that has the same parameters as their parent constructor.
- Use the methods from their parent class to add values for the properties.
- Each class (Bench, Chair and Sofa) should implement a *Printable* interface which contains print(), sneakpeek() and fullinfo() methods.
  - ➢ When called, the print method should print the name of the product, whether it is for sleeping and its area, for example:

    Desk, sitting only, 1000cm2
  - ➢ When called, the sneakpeek method should only print the name of the product, for example:

    Desk
  - ➢ When called, the fullinfo() method should print the name of the product, its width, length and height, its area and whether it is for sleeping or sitting only. For example:

    Desk, sitting-only, 1000cm2, width: 250cm, length: 40cm, height: 50cm

---

The name of the product is the same as the name of its class. To find it out use the get_class PHP function. It accepts an object as the first argument and returns the name of the class that object was instantiated from. You can also use it inside class by passing $this as a function argument. Here's an example:

```
class Person
{
    public function print()
    {
        echo get_class($this);
    }
}

$person = new Person();
echo get_class($person); //Person
$person->print(); //Person
```

Grading:

Part 1 of the exercise:	**1** point max
Part 1 and **2** of the exercise:	**3** points max
Part 1, **2** and **3**:	**5** points max

Deadline:

One week after the day of presentation (23:59).
