EREDITARIETÀ

L'ereditarietà' e' un concetto fondamentale nell'ObjOriented
Si possono creare classi figlie di un'altra classe.

```php
class Person{
        public $name;
        public $surname;
        public $age;
        public $height;
        public static $count =0;

        public function __construct($nome, $cognome, $eta){
                $this->name = $nome;
                $this->surname = $cognome;
                $this->age = $eta;
                $this->height = rand(160, 200);
                self::$count++;
        }

class Teacher extends Person{
        public$salary;
        public $subject;

        public function __construct($nome, $cognome, $eta, $salario, $materia){
                parent::__construct($nome, $cognome, $eta);
                $this->salary = $salario;
                $this->subject = $materia;

        }
}
}

$francesco = new Teacher("Francesco", "Talamona", "eta", 35, 3000, "Php");
print_r($francesco);

//return Teacher Object
//(
//      [salary] => 3000
//      [subject] => Php
//      [name] => Francesco
//      [surname] => Talamone
//      [age] => 35
//      [height]
//)
```

class Teacher extends Person{ } con extends dichiariamo la classe y figlia di x e al suo interno dichiariamo le proprietà' con la stessa sintassi della classe padre.

__construct istruiamo la classe allo stesso modo del padre ma con nuovi parametri che solo il figlio avrà'

parent::__construct($nome, $cognome, $eta); diciamo alla funzione di prendere dalla classe padre questi parametri nel modo spiegato li'

ABSTRACT

Una classe astratta è definibile come un particolare tipo di classe che **non può essere istanziata**, cioè non è possibile creare un oggetto da una classe astratta, ma può solo essere estesa da un'altra classe.
E' possibile aggiungere uno o più **metodi astratti** i quali dovranno essere (necessariamente) ridefiniti da una sottoclasse per poter essere utilizzati.

DEPENDENCY INJECTION

Una volta create le classi astratte e relative sottoclassi possiamo creare delle classi a parte che le richiamano.
E' possibile richiamarle tramite la Dependency Injection.

COMPOSITION

La composition ci permette di unire tutta la struttura creata con la Dependency Injection per poter creare infinite classi e poterle gestire.

```php
abstract class NomeClasseAstratta1{
     abstract public function nomeFunzioneAstratta();
}

class nomeSubclasse extends nomeClasseAstratta{
     public function nomeFunzioneAstratta(){
     }
}


class nomeClasse{
       public $variabile1;

       public function __construct(NomeClasseAstratta1 $parametroFormale1){
              $this->variabile1 = $parametroFormale1;
       }

       public function nomeFunzione1(){
              $this->variabile1->nomeFunzioneAstratta();
       }
}


$nomeVariabile = new nomeClasse(new nomeSubclasse());

$nomeVariabile->nomeFunzione1();
```