

Тестирование теоретических знаний

1. Что выведется на экран?

```
class ClassA {}  
  
$a = new ClassA();  
$b = new ClassA();  
  
var_dump($a === $b);
```

Варианты ответов:

1. bool(false)
2. bool(true)
3. Вылетит ошибка.

2. Что выведется на экран?

```
var_dump([  
  
"a" => 1,  
  
"b" => 2,  
  
"c" => [1,2,3]  
  
] === [  
  
"a" => 1,  
  
"b" => 2,  
  
"c" => [1,2,3]  
  
]);
```

Варианты ответов:

1. bool(false)
2. bool(true)
3. Вылетит ошибка.

3. Что выведется на экран?

```
class ClassA {  
  
public int $var1;  
  
public function __construct(int $var1)  
  
{  
  
$this->var1 = $var1;  
  
}  
  
}  
  
$a = new ClassA(5);  
$b = $a;  
$b->var1 = 10;  
echo $a->var1;
```

Варианты ответов:

1. 5
2. 10
3. Вылетит ошибка.

4. Что выведется на экран?

```
class ClassA {  
  
public int $var1;
```

```

public function __construct(int $var1)
{
    $this->var1 = $var1;
}
}

$a = new ClassA(5);
$b = $a;
$a->var1 = 10;
$a = null;
echo $b->var1;

```

Варианты ответов:

1. 5
2. 10
3. Вылетит ошибка.

5. Что выведется на экран?

```

class ClassA {
    public int $var1;
    public function __construct(int $var1)
    {
        $this->var1 = $var1;
    }
}

$a = new ClassA(5);
$b = &$a;
$b->var1 = 10;
$a = null;
echo $b->var1;

```

Варианты ответов:

1. 5
2. 10
3. Вылетит ошибка.

6. Что выведется на экран?

```

$numbers = [1,2,3,4,5];
foreach ($numbers as $number) {
    $number **= 2;
}
echo $numbers[4];

```

Варианты ответов:

1. 5
2. 10
3. 25
4. Вылетит ошибка.

7. Что выведется на экран?

```
$numbers = [1,2,3,4,5];  
foreach ($numbers as &$number) {  
    $number **= 2;  
}  
$number = 10;  
echo $numbers[4];
```

Варианты ответов:

1. 5
2. 10
3. 25
4. Вылетит ошибка.

8. Что выведется на экран?

```
interface Movable {  
    public function move(): void;  
}  
  
interface Breathable {  
    public function breath(): void;  
}  
  
class Animal implements Movable, Breathable {  
    public function move(): void  
    {  
        echo self::class . ' move';  
    }  
}  
  
class Dog extends Animal {  
    public function move(): void  
    {  
        echo self::class . ' move';  
    }  
}  
  
(new Dog())->move();  
echo "\n";  
  
(new Animal())->move();
```

Варианты ответов:

1. Dog move
2. Animal move
3. Dog move
4. Dog move
5. Animal move
6. Animal move
7. Вылетит ошибка.

9. Каков результат выполнения данного кода?

```
interface Movable {  
    public function move(): void;  
}
```

```

interface Breathable {
public function breath(): void;
}

interface Dog extends Movable, Breathable {
public function bark(): void;
}

```

Варианты ответов:

1. Выполнится успешно.
2. Вылетит ошибка.

10. Что выведется на экран?

```

interface Movable {
public function move(): void;
}

interface Breathable {
public function breath(): void;
}

class Animal implements Movable, Breathable {
public function move(): void
{
echo static::class . ' move';
}

public function breath(): void
{
echo static::class . ' breath';
}
}

class Dog extends Animal {
}

$dog = new Dog();
$dog->move();

echo "\n";

$animal = new Animal();
$animal->move();

```

Варианты ответов:

1. Dog move
2. Animal move
3. Dog move
4. Dog move
5. Animal move
6. Animal move
7. Вылетит ошибка.

11. Что выведется на экран?

```

interface Movable {

```

```

public function move(): void;
}

interface Breathable {
public function breath(): void;
}

class Animal implements Movable, Breathable {
public function move(): void
{
echo self::class . ' move';
}

public function breath(): void
{
echo self::class . ' breath';
}
}

class Dog extends Animal {
}

$dog = new Dog();
$dog->breath();

echo "\n";

$animal = new Animal();
$animal->breath();

```

Варианты ответов:

1. Dog breath
2. Animal breath
3. Dog breath
4. Dog breath
5. Animal breath
6. Animal breath
7. Вылетит ошибка.

12. Что делает composer install команда?

1. Установит все зависимости согласно composer.json в vendor директорию.
2. Установит все зависимости согласно composer.json (даже если есть composer.lock файл) в vendor директорию, после обновит composer.lock.
3. Установит все зависимости согласно composer.lock (если есть) в vendor директорию, иначе использует composer.json и после создаст composer.lock файл.
4. Установит все зависимости согласно composer.lock (если есть) в vendor директорию, иначе использует composer.json.

13. Что делает composer update команда?

1. Установит все зависимости согласно composer.json в vendor директорию.
2. Установит все зависимости согласно composer.json (даже если есть composer.lock файл) в vendor директорию, после обновит composer.lock.
3. Установит все зависимости согласно composer.lock (если есть) в vendor директорию, иначе использует composer.json и после создаст composer.lock файл.
4. Установит все зависимости согласно composer.lock (если есть) в vendor директорию, иначе использует composer.json.

14. Какая composer команда установит новый пакет и добавит его в composer.json?

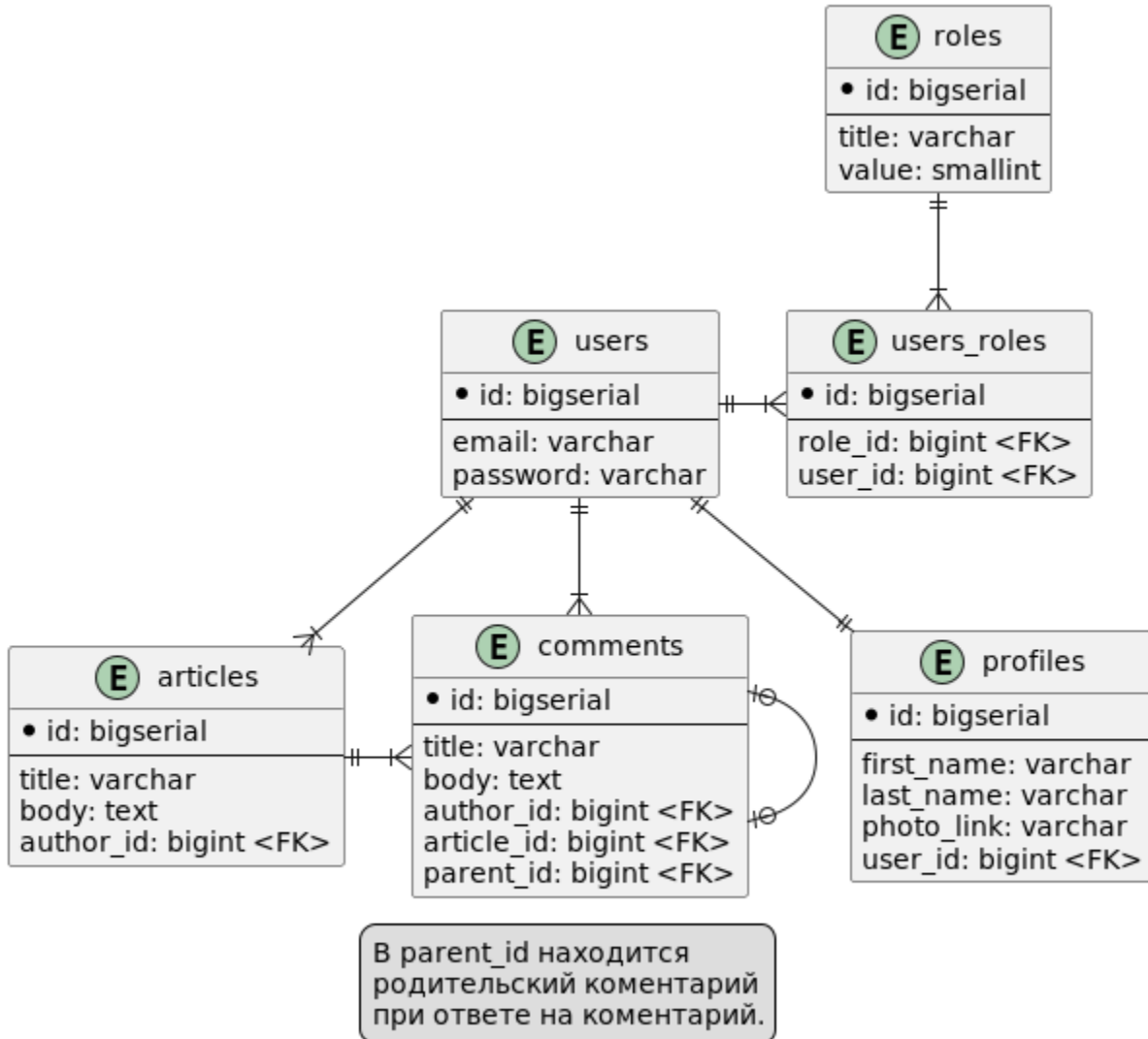
1. composer require <имя пакета>
2. composer install <имя пакета>

3. composer update <имя пакета>
4. composer add-dependency <имя пакета>

15. Какой сегмент (Обозначено как X) в версии композер пакета отвечает за версии без добавления нового функционала и с сохранением обратной совместимости (без изменений интерфейса)?

1. X.2.3
2. 1.X.3
3. 1.2.X

Используя имеющуюся выше диаграмму напишите такие sql запросы



16. Достаньте все комментарии для статьи с id = 101, нужно достать такие столбцы:

1. article_id
2. article_title
3. article_body
4. comment_id
5. comment_title
6. comment_body
7. comment_parent_id

17. Достаньте данные пользователя и все роли для пользователя с id = 256, нужно достать такие столбцы:

1. user_id
2. email
3. first_name
4. last_name
5. photo_link
6. role_title

18. Добавьте пользователю с id 225 новую роль (id роли = 8).
19. Обновите photo_link для профиля пользователя с id = 67.
20. Удалите пользователя с id = 78 со всеми его связями.