## Задача №1

Имеется база со следующими таблицами:

CREATE TABLE `users` (

    `id`         INT(11) NOT NULL AUTO\_INCREMENT,

    `name`       VARCHAR(255) DEFAULT NULL,

    `gender`     INT(11) NOT NULL COMMENT '0 - не указан, 1 - мужчина, 2 - женщина.',

    `birth\_date` INT(11) NOT NULL COMMENT 'Дата в unixtime.',

    PRIMARY KEY (`id`)

);

CREATE TABLE `phone\_numbers` (

    `id`      INT(11) NOT NULL AUTO\_INCREMENT,

    `user\_id` INT(11) NOT NULL,

    `phone`   VARCHAR(255) DEFAULT NULL,

    PRIMARY KEY (`id`)

);

Напишите запрос, возвращающий имя и число указанных телефонных номеров девушек в возрасте от 18 до 22 лет.  
Оптимизируйте таблицы и запрос при необходимости.

## Задача №2

Имеется строка:

<https://www.somehost.com/test/index.html?param1=4&param2=3&param3=2&param4=1&param5=3>

Напишите функцию, которая:

1. удалит параметры со значением “3”;
2. отсортирует параметры по значению;
3. добавит параметр url со значением из переданной ссылки без параметров (в примере: /test/index.html);
4. сформирует и вернёт валидный URL на корень указанного в ссылке хоста.

В указанном примере функцией должно быть возвращено:

<https://www.somehost.com/?param4=1&param3=2&param1=4&url=%2Ftest%2Findex.html>

## Задача №3

Напишите код в парадигме ООП, соответствующий следующей структуре.

**Сущности:** Пользователь, Статья.  
**Связи:** Один пользователь может написать несколько статей. У каждой статьи может быть только один пользователь-автор.

Функциональность:

* возможность для пользователя создать новую статью;
* возможность получить автора статьи;
* возможность получить все статьи конкретного пользователя;
* возможность сменить автора статьи.

Если вы применили какие-либо паттерны при написании, укажите какие и с какой целью.  
Код, реализующий конкретную функциональность, не требуется, только общая структура классов и методов.  
Код должен быть прокомментирован в стиле PHPDoc.

## Задача №4

Проведите рефакторинг, исправьте баги и продокументируйте в стиле PHPDoc код, приведённый ниже (таблица users здесь аналогична таблице users из задачи №1).  
Примечание: код написан исключительно в тестовых целях, это не "жизненный пример" :)

function load\_users\_data($user\_ids) {

    $user\_ids = explode(',', $user\_ids);

    foreach ($user\_ids as $user\_id) {

        $db = mysqli\_connect("localhost", "root", "123123", "database");

        $sql = mysqli\_query($db, "SELECT \* FROM users WHERE id=$user\_id");

        while($obj = $sql->fetch\_object()){

            $data[$user\_id] = $obj->name;

        }

        mysqli\_close($db);

    }

    return $data;

}

// Как правило, в $\_GET['user\_ids'] должна приходить строка

// с номерами пользователей через запятую, например: 1,2,17,48

$data = load\_users\_data($\_GET['user\_ids']);

foreach ($data as $user\_id=>$name) {

    echo "<a href=\"/show\_user.php?id=$user\_id\">$name</a>";

}

Плюсом будет, если укажете, какие именно уязвимости присутствуют в исходном варианте (если таковые, на ваш взгляд, имеются), и приведёте примеры их проявления.

## Вопрос №1

Занимались ли вы разработкой автоматизированных функциональных тестов, нагрузочных тестов, либо unit-тестов? Если да – сообщите, что доводилось использовать, какие решали задачи и с каким результатом. Если нет – не страшно, поможем освоить :)

## Вопрос №2

Перечислите книги (либо веб-ресурсы), которые Вы прочитали и усвоили (либо регулярно читаете), и какие можете рекомендовать своим коллегам для использования в работе, повышения профессионализма, расширения кругозора.