Задача на PHP

Описать классы и реализацию

Игрок (Player)
1. имя. Приватное свойство.

```php
$player1 = new Player("Петя");
```

2. Его город. Приватное свойство.

```php
$player1->setCity("Минск")
```

Турнир (Tournament) - показывает в каких турнирах участвует игрок
1. имя. Приватное свойство.

```php
$tournament1 = new Tournament("Турнир А");
```

2. Турнир может иметь дату начала. Если она не передана, то датой начала считается завтрашний день. Передать дату можно вот так (год.месяц.день):

```php
$tournament1 = new Tournament("Турнир А", “2021.10.25”);
```

3. В турнире находятся игроки. Игроки хранятся в приватном свойстве. Игроков в турнир нужно добавить так

```php
$tournament1
 ->addPlayer($team1)
 ->addPlayer($team2);
```

4. Необходимо реализовать метод createPairs, который для турнира создаст пары игроков так, чтобы каждый игрок встретился с каждым. Результатом работы метода будет вывод пар, которые будут встречаться друг с другом в определенный день с датой игры в формате день-месяц-год. Один игрок не может играть более 1 игры в день. Турнир необходимо провести максимально быстро.

```php
$tournament1->createPairs();
```

Пример: есть 2 игрока:
имя: Петя, город: Минск
имя: Вася, город: Могилев
Результат работы createPairs
Турнир А, 01.01.2021
Петя (Минск) - Вася (Могилев)

для 3 игроков
Турнир А, 01.01.2021
Петя - Вася
Турнир А, 02.01.2021
Петя - Коля
Турнир А, 03.01.2021
Вася - Коля

Игроков может быть сколько угодно, четное или нечетное количество

Пример алгоритма реализации для 8 игроков. Следовать алгоритму необязательно, можно сделать по-своему.
1 день
Записываем игроков следующим образом
1 2 3 4
8 7 6 5

Пары игроков (одна колонка выше - 1 пара)
1 - 8
2 - 7
3 - 6
4 - 5

далее двигаем игроков по часовой стрелке, не двигая первого. Т.е. Для второго дня

1 *8 2 3*
7 6 5 4

Пары игроков
1 - 7
8 - 6
2 - 5
3 - 4

Для третьего и последующих дней продолжаем движение по кругу, не двигая игрока 1. Если число игроков нечетное, то один из игроков должен пропустить день. Это можно сделать введением игрока-заглушки, которая доведет число игроков до четного.

Результат задачи:

1. Файл index.php, который создает игроков и турниры, заполняет их данными, выводит результат. В конце документа Модифицировать его нельзя, используйте как есть
2. В файле index.php вначале подключаются 2 файла: player.php, tournament.php. Их вам нужно реализовать самостоятельно, в один файл положить реализацию класса Player, в другой Tournament.

Перед передачей задачи нам, запустите index.php и проверьте соответствует ли результат его работы ожидаемому результату, который описан в комментариях в index.php.

Задача на БД

Написать как могла бы выглядеть структура данных для PHP задачи выше

Базовая структура
Таблица Player
- id, первичный ключ
- name
- city

Таблица Tournament
- id, первичный ключ
- name
- start_date

1. Написать SQL по созданию базовой структуры

2. Реализовать возможность создания связи многие ко многим для 2 таблиц выше (у одного игрока может быть много турниров, у одного турнира может быть много игроков)

3. написать INSERT запросы на создание 3 турниров. Состав турниров.
Tournament A
Player 1
Player 2

Tournament B
Player 1
Player 2
Player 3

Tournament C
Player 3
Player 4
Player 5

4. Написать один SQL запрос, который выведет турниры с игроками внутри
Пример результата (2 колонки)
Tournament A Player 1
Tournament A Player 2
Tournament B Player 1
Tournament B Player 2
Tournament B Player 3
Tournament C Player 3
Tournament C Player 4
Tournament C Player 5

```php
<?php

require_once "player.php"; //внутри реализовать class Player
require_once "tournament.php"; //внутри реализовать класс Tournament

$tournamentA = new Tournament("Tournament A", "2022.12.30");
$tournamentA
        ->addPlayer( (new Player("Player 1"))->setCity("Minsk") )
        ->addPlayer( (new Player("Player 2"))->setCity("Mogilev") )
        ->addPlayer( (new Player("Player 3"))->setCity("Vitebsk") )
        ->addPlayer( (new Player("Player 4"))->setCity("Gomel") );
$tournamentA->createPairs();
/*
ожидаемый результат выполнения $tournamentA->createPairs();
Примечание. Все пары уникальные.
Tournament A, 31.12.2022
Player 1 (Minsk) - Player 4 (Gomel)
Player 2 (Mogilev) - Player 3 (Vitebsk)
Tournament A, 01.01.2023
Player 1 (Minsk) - Player 3 (Vitebsk)
Player 4 (Gomel) - Player 2 (Mogilev)
Tournament A, 02.01.2023
Player 1 (Minsk) - Player 2 (Mogilev)
Player 3 (Vitebsk) - Player 4 (Gomel)
*/
$tournamentB = new Tournament("Tournament B");
$tournamentB
        ->addPlayer( new Player("Player 1" ) )
        ->addPlayer( new Player("Player 2" ) )
        ->addPlayer( new Player("Player 3" ) )
        ->addPlayer( new Player("Player 4" ) )
        ->addPlayer( new Player("Player 5") )
        ->addPlayer( new Player("Player 6") )
        ->addPlayer( new Player("Player 7") );

$tournamentB->createPairs();

/*
ожидаемый результат выполнения $tournamentB->createPairs();
Примечание. Все пары уникальные. В каждом туре есть одна игрок его пропускающий. Например, в первом туре это Player 1.
Дата в примерах ниже из расчете, что сегодня 04.05.2022
Tournament B, 05.05.2022
Player 2 - Player 7
Player 3 - Player 6
Player 4 - Player 5
Tournament B, 06.05.2022
Player 1 - Player 7
Player 2 - Player 5
Player 3 - Player 4
Tournament B, 07.05.2022
Player 1 - Player 6
Player 7 - Player 5
Player 2 - Player 3
Tournament B, 08.05.2022
Player 1 - Player 5
Player 6 - Player 4
Player 7 - Player 3
Tournament B, 09.05.2022
Player 1 - Player 4
Player 5 - Player 3
Player 6 - Player 2
Tournament B, 10.05.2022
Player 1 - Player 3
Player 4 - Player 2
Player 6 - Player 7
Tournament B, 11.05.2022
Player 1 - Player 2
Player 4 - Player 7
Player 5 - Player 6
*/
```