```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Parallel</title>
</head>

<body>

    <script>
        /*
,            ***** Parallel ****
,
,            비동기(Async) Parallel
,            우리가 두개의 API를 얻는다고하자. 하나는 기존의 영화이고 다른 하나를 또 해보겠다.
,            그리고 우리는 리스트를 얻고싶다.
,            인기있는 영화들과 곧 개봉하는 영화들.
,            이것들을 위해서 우리는 Promise all을 사용할것이고, destructuring assignment(구조분해 할당)을 할것이다
,
,            우리는 모든 Promise를 기다릴것이다.
,            const response = await Promise.all([
,                fetch("https://yts.mx/api/v2/list_movies.json"),
,                fetch("https://yts.mx/api/v2/movie_suggestions.json?movie_id=100")
,            ]).then(value => console.log(value));
,            첫번째와 두번째로 분리했다. 이제 알다시피 Promise all 그리고 then(value)
,            이 value는 이 두 Promise의 결과값의 array이다.
,
,            response를 사용하는 대신에 우리가 할 수 있는것은
,            destructuring을 상용하는것이다.
,            우리가 취할 첫번째 value는 movies, 두번쨰 value는 suggestions
,            const [ moviesResponse, suggestionsResponse ] = await Promise.all ...
,
,            이것들은 우리에게 Response를 줄것이다.
,            그리고 movies, suggestions을 선언하고 await promiseall([])을 해준다.
,            그 후에 콘솔으로 movies와 suggestions을 찍어보자.
,
,            try {
,                const [ moviesResponse, suggestionsResponse ] = await Promise.all([
,                    fetch("https://yts.mx/api/v2/list_movies.json"),
,                    fetch("https://yts.mx/api/v2/movie_suggestions.json?movie_id=100")
,                ]);
,                const [ movies, suggestions ] = await Promise.all([
,                    moviesResponse.json(),
,                    suggestionsResponse.json()
,                ]);
,
,                console.log(movies, suggestions)
,            }
,
,            보다시피 우리는 parallel(병렬)로 동작하고 movies와 suggestions를 가지고있다.
,
,            이것들은 Async await(비동기 대기)를 위한것이다.
,            이런식으로 Async Await를 사용ㅇ하기를.
,            하나의 문제는 함수안에 집어넣어야한다는점.
,
```

```
,            ** Axios
,
,            fetch가 이것들 중 최고라고 생각하지않는다. 여기에는 fetching해주는 많은 라이브러리들이 있는데
,            대표적으로 Axios가 있다.
,            나를 위해 모든것을 json이나 text로 변경하기 때문에
,            나는  이런것들을 할 필요가 없어질 수도있다.
,            const [ movies, suggestions ] = await Promise.all([
,                moviesResponse.json(),
,                suggestionsResponse.json()
,            ]);
,
,        */

         // async await
         const getMoviesAsync = async() => {
            try {
                const [ moviesResponse, suggestionsResponse ] = await Promise.all([
                    fetch("https://yts.mx/api/v2/list_movies.json"),
                    fetch("https://yts.mx/api/v2/movie_suggestions.json?movie_id=100")
                ]);
                const [ movies, suggestions ] = await Promise.all([
                    moviesResponse.json(),
                    suggestionsResponse.json()
                ]);

                console.log(movies, suggestions)
            } catch(e) {
                console.log(`❌ ${e}`);
            } finally {
                console.log("we are done");
            }
        }
        getMoviesAsync();
    </script>
</body>

</html>
```