```
<!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>arrow function in real world</title>
</head>
<body>
    <script>
        /*
,            ***** Array.prototype.find() *****
,
,            Array.prototype.find는 제공되는 테스트 조건을 만족하는
,            첫번째 엘리먼트 값을 리턴하는 함수.
,
,            const emails = ["bibi@no.com", "barnes@gmail.com",
"chrome@bonmi.com", "heartz@naver.com"];
,            예를들어 여기서 gmail.com인 주소를 찾는다고해보자
,
,            const foundMail = emails.find()
,            몇개의 argument가 필요하다.
,
,            element는 현재 처리하고있는 element를 나타내고
,            index는 현재 element의 1,2,3,4같은 index를 나타내고,
,            array는 array,find 함수가 호출한 array를 나타낸다.
,
,            arrowfunction으로 무언가(조건)를 return할것인데,
,            만약 조건이 true면 그 말은 우리가 item을 찾았다는것.
,
,            const foundMail = emails.find(item => item.includes("@gmail.com"))
,            console.log(foundMail)  출력값은 barnes@gmail.com 이 출력된다.
,            string인 item에 includes를 해줄것인데,
,            includes는 string을 찾아준다.
,            또한, includes는 true나 false를 return.
,            찾았으면 true를 return하고 못찾으면 false를 return
,
,            ***** Array.prototype.filter() *****
,
,            filter메소드는 제공된 함수의 조건을 만족한 모든 엘리먼트로 새로운 array를 만든다.
,            그러니 첫번째 엘리먼트만이 아닌 모든 엘리먼트를 반환한다.
,
,            element,index,array는 find와 같은맥락.
,            * element가 true를 반환하는 것들만 array에 넣고 false를 반환하는 element는 탈락
,
,
,            ***** Array.prototype.forEach() *****
,
,            forEach 메소드는 각 array의 엘리먼트 마다 제공된 함수를 실행한다. (map과 비슷한 맥락)
,            예를 들어 @어쩌고.com에는 관심이없고 만약 username만 가져오고 싶다고 해보자.
,
,            split()은 어떤 조건으로 string을 나눠서 하나의 array에 두개의 element로 만들어준다.
,            const name = "buucky barnes"
,            name.split(" ")
,            ["bucky", "barnes"] 문자열을 조건으로도 넣을 수 있다.
,
,            하단 예제에서는 array에서 반 (@를 기준으로 앞쪽부분)만 가져오고싶으니
,            [0] 인덱스 0번을 가지고오면 된다.
,
,            * forEach는 array의 각 element마다 함수를 실행시키는데,
,            만약 변수가 있으면 어떻게 될까.
```

```
,
,            const cleaned = [];
,            emails.forEach(email => {
,                console.log(email.split("@")[0]) // array에서 첫번째것들만 가지고옴
,                cleaned.push(email.split("@")[0]);
,            });
,            console.log(cleaned);
,
,            반환값은 ['bibi@no.com', 'barnes@gmail.com', 'chrome@bonmi.com',
'heartz@naver.com']가 되지만
,            너무 길다 이런때에 map 을 사용
,
,            ***** Array.prototype.map() *****
,
,            map은 forEach지만 반환된 element들로 새로운 array를 만들어준다.
,
,            ** arrow function
,
,            코드를 보기 좋게 만들고 이렇게 callback 함수를 사용하는 기능들을 사용할 때
,            한줄로 되면 훨씬 읽기 쉬울것이다
,            function을 만들고 ()도 넣고 {}를 추가하는것보다.
,            implicit return때문에 더 편해진것도 있다.
,
,            **
,
,            const clean = emailsTwo.map(email => email.split("@")[0]);
,            무언가를 return할때에 값을 return하는 이경우가 아니라 object를 return하고 싶다면 어
떻게 될까.
,            예를들어 name만 return하는것이 아니라 그 순서도 return 하고싶다면.
,
,            const cleanObject = emailsTwo.map((email,index) => ({
,                username : email.split("@")[0],
,                // index : index 이것은 es6문법으로 하면 로도 가능.
,                index
,            }));
,
,            우리는 {}를 써서 implicit return을 못쓰게 만든다.
,            그래서 ()를 추가해준다.
,            ()를 추가하면 함축적으로 이 object를 return하겠다는 뜻이 된다.
,
,         */
        const emails = [
            "bibi@no.com",
            "barnes@gmail.com",
            "chrome@bonmi.com",
            "heartz@naver.com"
        ];

        const emailsTwo = [
            "bibi@no.com",
            "barnes@gmail.com",
            "chrome@bonmi.com",
            "heartz@naver.com"
        ]

        /*
,            find
,            const foundMail = emails.find(item => true);
,            console.log(foundMail)
,            이런식으로 하면 bibi@no.com이 출력된다. 왜냐하면 true이기 때문에 첫번째 아이템이 바로
리턴
,         */
```

```
        const foundMail = emails.find(item => item.includes("@gmail.com"))
        console.log(foundMail, "this is find")

        /*
,            filter
,            potato나 item은 내가 정하는 변수명 , gmail.com을 가지고있지 않는 것들로 array
,         */
        const noGmail = emails.filter(potato => !potato.includes("@gmail.com"))
        console.log(noGmail, "this is filter")

        /*
,            forEach
,            emails.forEach(email => email.split(" ")); 이런식으로 하면 array하나만 반환한
다.
,            email.split("@")[0] // array에서 첫번째것들만 가지고옴
,         */
        emails.forEach(email => {
            email.split("@")[0]
        });
        console.log(emails, "this is forEach");

        /*
,            map
,            map은 array를 반환할것이니까 저장할 곳을 만듬.
,         */
        const clean = emailsTwo.map(email => email.split("@")[0]);
        console.log(clean, "this is map");

        /*
,            map에서 object로 return 받고싶은 경우
,         */
        const cleanObject = emailsTwo.map((email,index) => ({
            username : email.split("@")[0],
            // index : index 이것은 es6문법으로 하면 로도 가능.
            index
        }));
        console.table(cleanObject); // console에서 table형식으로 볼 수있게해줌

    </script>
</body>
</html>
```