# Quiz

Test your knowledge on generic types with this quiz.

## Working with generic types

1

We have a type within our codebase for a comment:

```
type Comment = {
  comment: string;
  email: string;
}
```

What generic type can we use with `Comment` that would create a type equivalent to the type below:

```
type ReadonlyComment = {
  readonly comment: string;
  readonly email: string;
}
```

2

The function below counts the occurrences of an item in an array:

```
function countDisinct(itemToCount, array) {
  return array.filter(item => item === itemToCount).length
}
```

How can we make this strongly-typed?

3

We need to create a function to remove occurrences of an item from an array. It needs to be strongly-typed and work with arrays containing any primitive type. Here's our attempt:

```
function remove<ItemType>(itemToRemove: ItemType, array: Arra
y<ItemType>): ItemType {
  return array.filter(item => item !== itemToRemove);
}
```

Can you identify the error in the above code?

4

We have the following interface which represents a user:

```
interface User {
  id: any;
  name: string;
  email: string
}
```

How can we improve this by removing the `any` type on the `id` property and letting the consumer of the interface supply its type:

5

We need to extend the generic list class we created in the last lesson. As a reminder, here's the class:

```
class List<ItemType> {
    private items: ItemType[] = [];

    add(item: ItemType) {
        this.items.push(item);
    }
}
```

We need to add a method called `getNth` which returns the array item at the *nth* position in the list. How could we implement this?

Check Answers

Well done!

Now that we have a solid understanding of how to create TypeScript types, we are going to start to use them in React components in the next chapter.