ВСТРОЕННЫЕ ОБОБЩЕНИЯ

В TS существует довольно много **встроенных дженериков**, часть из которых для внутренних нужд языка, а часть для рутинных задач в разработке. Первый, с которым мы знакомились - это дженерик, запрещающий менять данные и его разновидность ReadonlyArray:

```ts
const arr: Array<number> = [1, 2, 3]; // Обычный массив

const roarr: ReadonlyArray<string> = ["dsds"]; // Неизменяемый массив
```

```ts
interface IState {
    data: {};
    tag: string;
}

function action(state: Readonly<IState>) {
    state.data = ""; // Error
}
```

В варианте выше Readonly дженерик запрещает изменения только на **первом уровне вложенности** в объекте. Свойства внутри state.data менять уже можно

Дженерик **Partial** добавляет всем свойствам объекта модификатор вопросительного знака (optional), делая их **необязательными:**

```ts
interface IState {
    data: {
        name: string;
    };
    tag: string;
}

const state: Partial<IState> = {
    data: {
        name: "John",
    }
};
```

Дженерик **Required** - это полная противоположность **Partial**. Он берет объект и удаляет у всех свойств модификатор необязательности (optional), делает все поля **обязательными**:

```ts
interface IState {
    data: {
        name: string;
    };
    tag?: string;
}

const strictState: Required<IState> = {
    data: {
        name: "John",
    }
    // Error, не хватает свойства tag
};
```

Строго говоря, эти дженерики **более правильно называть типами**, но на уровне кода вы уже знаете что это такое. Большую часть из них мы изучим дальше в уроках