**1. Переменные:**

let username: string = "Alice";

let age: number = 30;

let isActive: boolean = true;

let allType: any = ‘любой тип’;

let unknown: unknown = ‘Тип не определен’;

**2. Объекты**

Interface People = {}

или

type People = {

readonly name: string, readonly не позволяет изменять значение после присвоения

age: number

city?: string - ? указывает, что свойство не обязательное

[key: number]: Boolean –объект может иметь любое количество (или отсутствовать) свойств типа number со знач Boolean

}

const people: People = {

name: “Maksim”, people.name = “Maks” – ошибка из-за readonly

age: 30,

city: “Perm” – можно не указывать, ошибки не будет

33: true – можно не указывать, ошибки не будет

}

|  |  |
| --- | --- |
| interface | type |
| Определение и использование | |
| - Используется для описания объектов, классов и других структур данных.  - Может расширять другие интерфейсы с помощью ключевого слова `extends`.  - Может быть реализован классами с помощью ключевого слова `implements`. | - Используется для определения алиасов типов. Это может быть, как объект, так и примитивные типы, объединения, пересечения и т.д.  - Не поддерживает наследование, как интерфейсы, но может использовать объединение и пересечение типов. |
| Расширяемость | |
| - Интерфейсы могут быть объявлены несколько раз, и TypeScript объединит их в один интерфейс. Это позволяет расширять интерфейсы в разных местах кода. | - Типы не могут быть объявлены несколько раз. Если вы попытаетесь повторно объявить один и тот же алиас типа, TypeScript выдаст ошибку. |
| Типы данных | |
| Чаще используется для описания структур объектов и классов. | - Более универсален и может использоваться для описания не только объектов, но и объединений, пересечений, примитивов и других типов. |

**3. Массивы:**

   typescript

   let numbers: number[] = [1, 2, 3, 4];

   let numberAndString: (number | string)[] = [1, 2, 3, 4, “5”];

   let names: Array<string> = ["Alice", "Bob", "Charlie"];

**4. Массив объектов:**

let peopleArr: People[] = [{name: “Alex”, age: 15}, {name: “Petr”, age: 22, city: “Moscow”}]

peopleArr.push({name: “Kris”, age: 25})

**5. Массивы с типом ReadonlyArray**

Это специальный массив, который нельзя изменять после его создания. Он используется для обеспечения неизменяемых данных, что позволяет избежать ошибок

let peopleArray: readonly number[] = [1, 2, 3, 4, 5]

let peopleArray: readonly People[] = [{name: “Alex”, age: 15}, {name: “Petr”, age: 22, city: “Moscow”}]

Чтобы добавить новый элемент в массив необходимо использовать spread оператор

Например:

let people2: People = {name: “Masha”, age: 30}

let peopleArray2: readonly People[] = […peopleArray, people2]

ReadonlyArray используют для работы с React и Angular (они чаще всего используют мутабельную структура данных)

**6. Функции:**

void – функция нечего не возвращает

   function greet(name?: string): string {

       return `Hello, ${name}!`;

   }

   const result: string = greet("Bob");

**7. Объединение типов:**

let value: string | number;

value = "Hello"; // допустимо

value = 42; // также допустимо