## 一、简答题

### 1.交叉类型和联合类型的标识分别是什么？

|  |
| --- |
|  |

### 2.类型别名和接口的异同？

|  |
| --- |
|  |

### 3.实现类型保护的方式有哪些？

|  |
| --- |
|  |

### 4.什么是索引类型?

|  |
| --- |
|  |

## 二、分析题

### 1.补全代码,使打印结果显示20

|  |
| --- |
| class User {    public nickname: string | undefined    public group: number = 20  }  class Log {    public count: number = 10    public keyword: string | undefined  }  function typeGuard(arg: User | Log):number {    if (请补全代码) {       return arg.group    }    if (请补全代码) {      return arg.count    }  }  let user = new User();  console.log(typeGuard(user)) // 输出结果20 |

### 2.下面代码报错，怎么改造代码，使运行结果正确

interface Cat {

    name: string

    miao:() => string

  }

interface Dog {

    name: string

    wang: () => string

  }

  const Animal1 = (): Cat | Dog => {

     return {

          name: "",

          miao: ():string => {return "2333"}

      }

  }

  let obj1 = Animal1();

  obj1.miao()  //报错

### 3.下面代码的执行结果分别是

type strType=string|number|boolean;

var str:strType;

str = true; // ?

str = "字符串" // ?

str = 12 // ?

str = {} // ?

### 4. 下面代码的执行结果分别是

type Name = string;

type NameResolver = () => string;

type NameOrResolver = Name | NameResolver;

function getName(n: NameOrResolver): Name {

　　if (typeof n === 'string') {

　　　　return n;

　　} else {

　　　　return n();

　　}

}

console.log(getName('aaa'))  // ?

console.log(getName(function () { return "张三" })) // ？

console.log(getName(function () { console.log(11)}))

// ?

### 5. 下面代码报错，怎么改造代码，使运行结果正确

interface Motorcycle {

  vType: "motorcycle";

  make: number;

}

interface Car {

  vType: "car";

  transmission: number;

}

interface Truck {

  vType: "truck";

  capacity: number;

}

type Vehicle = Motorcycle | Car | Truck;

function evaluatePrice(vehicle: Vehicle) {

  return vehicle.capacity \* Math.PI;

}

const myTruck: Truck = { vType: "truck", capacity: 9.5 };

console.log(evaluatePrice(myTruck));