## 简答题

### 请简述枚举有几种类型，分别是什么，语法写出来

|  |
| --- |
| 1.数字枚举  enum Direction {  NORTH,  SOUTH,  EAST,  WEST,  }  2.字符串枚举  enum Direction {  NORTH = "NORTH",  SOUTH = "SOUTH",  EAST = "EAST",  WEST = "WEST",  }  3.异构枚举  enum Enum {  A,  B,  C = "C",  D = "D",  E = 8,  F,  } |

### 请简述数字枚举的注意点是什么？

|  |
| --- |
| **enum** Gender {    Male,    Female  }  1.如果是给Male赋值，那么后面的值都改变  2.如果是给Female赋值，那么Male不改变，Female的值发生改变  3.也可以给Male和Female分别赋值  4.数字枚举不仅仅可以使用默认值，还可以使用常量，计算后的结果 |

### 请简述字符串枚举的注意点是什么？

|  |
| --- |
| 1.如果使用字符串给前面的枚举值赋值了字符串，那么后面的也必须赋值（无论是字符串还是数字都可以）  2.如果使用字符串给前面的枚举值赋值了字符串，那么后面的也必须赋值，如果不赋值，那么就使用默认值，也会被ts认为是字符串枚举  3.和数字枚举不一样，字符串枚举不能使用常量或者计算结果给枚举值赋值  4.虽然字符串枚举不能够使用常量或计算结果给枚举值赋值，但它可以使用内部的其他枚举值来赋值 |

### 普通枚举和常量枚举的区别？

|  |
| --- |
| 1.普通枚举会生成真实存在的对象  2.常量枚举不会生成真实存在的对象，而是利用枚举成员的值直接替换使用到的地方 |

## 运行题

### 1.下面代码的执行结果是:

|  |
| --- |
| enum Color {Red, Green, Blue}  let c: Color = Color.Green;  //运行结果  1 |

### 2.下面代码的执行结果是:

|  |
| --- |
| enum Color {Red = 1, Green, Blue = 4}  let c: Color = Color.Green;  // 运行结果  2 |

### 3.下面代码的执行结果是:

|  |
| --- |
| enum Color {Red = 1, Green, Blue}  let colorName1: string = Color[2];  console.log(colorName1) // 运行结果  Green  let colorName2: string = Color[Color.Blue];  console.log(colorName2) // 运行结果  Blue |

### 4.下面ts代码编译之后的js代码是什么

|  |
| --- |
| const enum Color {    Red,    Green,    Blue  }    let c: Color = Color.Green;  // 编译之后的js代码是什么  Var c=1 |

### 5.下面代码会有什么问题嘛？为什么？

|  |
| --- |
| enum Color {    Red,    Green = 'Green Color',    Blue  }  会，如果使用字符串给前面的枚举值赋值了字符串，那么后面的也必须赋值 |