## 简答题

### 请简述枚举有几种类型，分别是什么，语法写出来

|  |
| --- |
| 数字枚举  enum NumEnum {    A,    B,    C  }  字符串枚举  enum StrEnum {    A = "A",    B = "B",    C = "C"  }  异构枚举  enum DifEnum {    A,    B = "B"  } |

### 请简述数字枚举的注意点是什么？

|  |
| --- |
| 1.数字枚举默认值从0开始递增  2.设置其中一个为某个数字后，后面的枚举成员默认值根据这个数字递增  3.数字枚举不仅可以使用默认值，还可以使用常量，计算后的结果，但是使用后要注意后面其他成员的默认值必须亲自初始化 |

### 请简述字符串枚举的注意点是什么？

|  |
| --- |
| 1.字符串枚举不能实现反向映射  2.如果使用字符串中前面的枚举成员进行了赋值，后面的也必须亲自赋值  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.可以使用内部的其它枚举值来赋值 |

### 普通枚举和常量枚举的区别？

|  |
| --- |
| 普通枚举通过编译后，会产生一个实际的对象，枚举成员就是里面的键值对，所以我们也可以使用枚举成员的实际数据  但是常量枚举通过枚举后不会有产生任何实际的东西，使用枚举成员时，编译后会直接转为对应的枚举值  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到使用到的地方 |

## 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 |

let c = 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  let c = 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  Let c = 1; |

### 5.下面代码会有什么问题嘛？为什么？

|  |
| --- |
| enum Color {    Red,    Green = 'Green Color',    Blue  }  Blue报错，因为Green值设置为字符串后，Blue的默认值无法像之前一样默认递增，需要手动赋值 |