# 数据库部分

#### 请说出增删改查对应的语法：

* 1. 增加语句

insert into 表名(列名1,列名2,…..) values(值1,值2,值3…)

* 1. 修改语句

update 表名 set 列名=值,列名=值 where 列名 = 值 and/or 列名=值

* 1. 删除语句

delete from 表名 where 列名=值 and / or 列名 = 值

* 1. 查询语句

Select 列名 / \* from 表名 where列名 = 值 and/or 列名=值

#### 常见的分组聚合函数有哪些？

* sum(数值列) 求和
* avg(数值列) 平均值
* max(数值列) 最大值
* min(数值列) 最小值
* count 统计行数

#### 为什么需要分组，分组语句的语法关键字是什么？ 可以问一些具体的案例

1. 分组是为了按表中某列进行汇总数，一般都是为了聚合数据
2. 语法：
   1. Select 分组列名,sum(价格列) from 表 group by 类型列 --》按类型统计各个类型下商品总价格
   2. Select 分组列名,count(\*) from 表 group by 地区列 ---》统计各个地区下的总数

#### 排序的关键字是什么

排序：order by 列名 asc / desc asc代表升序，desc代表降序

#### 连接查询语句语法是什么？外连接和内连接的区别是什么？

内连接：Select \* from 表A inner join 表B on 表A.外键Id列 = 表B.主键Id列

左/右连接：select \* from 表A left/right join 表B on 表A.外键Id列 = 表B.主键Id列

目的：查询出两个表的所有数据，但是需要使用指定的列相等才会关联成一条完整记录

内连接必须条件一致才会连接成一条记录，而左右外连接如果没有匹配的记录，会用NULL填充一行数据显示。

#### 分页查询的语法：

1. Mysql中：limit关键字，limit 0 ,10 ---》从0开始，取10条
2. Sqlserver中：子查询配合top关键字实现
3. Oracle中：使用ROWNUM

#### 事务的概念和好处是什么？

事务是为了保证数据的安全性，同一个功能下的多条增删改语句，要么都成功提交要么都取消执行，前后保持数据的一致状态。

如转账操作：A给B转账，A账户需要修改减少余额，B账户需要修改增加余额，如果A账户的语句刚执行完毕，突然发生错误或者意外，则会导致B账户余额不能得到增加，此时使用事务将两个语句进行控制，要么都变化，要么都取消。

事务的执行：开始事务 start tran

提交事务 commit tran

回滚事务 rollback tran

#### 视图的概念和好处是什么？

视图是一个虚拟的表，其实就是一条复杂sql语句的缩影，每次都要写复杂的查询sql很麻烦，就创建视图指向该sql语句，以后再需要查询直接使用视图即可，如下

Create view vw\_xxx as select \* from 表。。。。。

以后如果使用时直接使用 select \* from vw\_视图名 则表示执行了对应的sql语句，简化了查询，提高了使用效率。

#### 存储过程的概念和好处

存储过程是指将某个功能对数据库的复杂操作逻辑，使用T-SQL直接写在数据库中，然后起一个调用的名字，还可以传递参数和返回数据，程序中直接调用改存储过程即可实现复杂的功能。

好处：

1. 安全性更高，因为程序只需要调用存储过程的名字，无需知道细节表及操作。
2. 效率更高，程序调用时的传输量变小，存储过程是预编译的，创建完毕后后续只需要执行，无需编译，提高了效率。
3. 模块化开发，数据库管理人员编写存储过程后，开发人员进行对接，开发效率和模块化更高

#### 什么是外键关联？

外键主要是和另外一个表的主键建立关联，外键列的数据只允许从对应主键表中的主键列中选取对应的数据，不能出现不合理的数据，保证了数据的规范性。

比如班级表和学生表，那么在学生表中创建一列对应班级的编号，该学生表的班级id则代表外键

#### 被外键关联的主表数据如何删除？

1. 先删除外键表关联数据，再删除主表数据，缺点是关联性太强，容易删除一些历史数据。
2. 软删除（增加一列是否删除的标识，执行删除时其实是更新该列的标识，查询时不显示则认为是删除）缺点是会积累大量无用数据。
3. 历史记录表，将删除的数据移动到历史记录表中。

#### 索引的作用？哪些列上适合创建索引？。

1. 索引是一种建立在表中列上用于提升查询速度的机制。
2. 经常查询、分组、排序的列适合创建索引，经常修改、数据重复的列上不适合。

#### 数据库的优化

1. sql尽量使用索引
2. 对sql语句优化
3. 子查询变成left join
4. limit 分布优化，先利用ID定位，再分页
5. or条件优化，多个or条件可以用union all对结果进行合并（union all结果可能重复）
6. 不必要的排序
7. where代替having,having 检索完所有记录，才进行过滤
8. 避免嵌套查询
9. 对多个字段进行等值查询时，联合索引
10. 尽量少用select \* ， 多加条件

#### varchar和char的使用场景

1. 用来存储字符
2. varchar适用字符长度经常变的
3. char适用字符长度固定的

# 前端部分

#### 你采用的是什么框架？

在前端中比较流行的有JQuery, Vue 等等

#### 标签上 alt与 title属性的区别是什么？

alt：搜索引擎识别，在图像无法显示时的替代文本；

title：元素注释信息，主要给用户解读。当鼠标放到文字或是图片上时有 title 文字显示

#### JS 的数据类型有哪些？

数据类型主要包括两部分：

基本数据类型：Undefined、Null、Boolean、Number 和 String

引用数据类型：Array 、Object

#### 简述下为何通过 ajax 发送的请求会出现乱码问题，如何解决？

编码格式冲突，直接在传输中文数据前加 encodURL() 编码

乱码的问题就是编码格式冲突，我们需要传输中文数据前面加一个 encodeURI()编码，  
例如：encodeURI($("#fk\_info").val())；在接受参数的页面对传过来的编码过后的内容用后端语言进行解码

#### 如何理解闭包？

1、定义和用法：当一个函数的返回值是另外一个函数，而返回的那个函数如果调用了其父函数内部的其它变量，如果返回的这个函数在外部被执行，就产生了闭包。

2、表现形式：使函数外部能够调用函数内部定义的变量。

#### 什么是跨域？

由于浏览器同源策略，凡是发送请求url的协议、域名、端口三者之间任意一与当前页面地址不同即为跨域。存在跨域的情况：

网络协议不同，如http协议访问https协议。

端口不同，如80端口访问8080端口。

域名不同，如qianduanblog.com访问baidu.com。

子域名不同，如abc.qianduanblog.com访问def.qianduanblog.com。

域名和域名对应ip,如www.a.com访问20.205.28.90.

#### 跨域请求资源的方法：

(1)、porxy代理

定义和用法：proxy代理用于将请求发送给后台服务器，通过服务器来发送请求，然后将请求的结果传递给前端。

实现方法：通过nginx代理；

注意点：1、如果你代理的是https协议的请求，那么你的proxy首先需要信任该证书（尤其是自定义证书）或者忽略证书检查，否则你的请求无法成功。

(2)、CORS 【Cross-Origin Resource Sharing】

定义和用法：是现代浏览器支持跨域资源请求的一种最常用的方式。

(3)、jsonp

定义和用法：通过动态插入一个script标签。浏览器对script的资源引用没有同源限制，同时资源加载到页面后会立即执行（没有阻塞的情况下）。

特点：通过情况下，通过动态创建script来读取他域的动态资源，获取的数据一般为json格式。

#### 谈谈垃圾回收机制

回收机制方式

1、定义和用法：垃圾回收机制(GC:Garbage Collection),执行环境负责管理代码执行过程中使用的内存。

2、原理：垃圾收集器会定期（周期性）找出那些不在继续使用的变量，然后释放其内存。但是这个过程不是实时的，因为其开销比较大，所以垃圾回收器会按照固定的时间间隔周期性的执行。

#### 开发过程中遇到的内存泄露情况，如何解决的？

1、定义和用法：

内存泄露是指一块被分配的内存既不能使用，又不能回收，直到浏览器进程结束。C#和Java等语言采用了自动垃圾回收方法管理内存，几乎不会发生内存泄露。我们知道，浏览器中也是采用自动垃圾回收方法管理内存，但由于浏览器垃圾回收方法有bug，会产生内存泄露。

2、内存泄露的几种情况:

(1)、当页面中元素被移除或替换时，若元素绑定的事件仍没被移除，在IE中不会作出恰当处理，此时要先手工移除事件，不然会存在内存泄露。

(2)、由于是函数内定义函数，并且内部函数--事件回调的引用外暴了，形成了闭包。闭包可以维持函数内局部变量，使其得不到释放。

#### 什么是盒子模型？

在网页中，一个元素占有空间的大小由几个部分构成，其中包括元素的内容（content），元素的内边距（padding），元素的边框（border），元素的外边距（margin）四个部分。这四个部分占有的空间中，有的部分可以显示相应的内容，而有的部分只用来分隔相邻的区域或区域。4个部分一起构成了css中元素的盒模型。

#### 简述一下src与href的区别

href 是指向网络资源所在位置，建立和当前元素（锚点）或当前文档（链接）之间的链接，用于超链接。

src是指向外部资源的位置，指向的内容将会嵌入到文档中当前标签所在位置；在请求src资源时会将其指向的资源下载并应用到文档内，例如js脚本，img图片和frame等元素。

当浏览器解析到该元素时，会暂停其他资源的下载和处理，直到将该资源加载、编译、执行完毕，图片和框架等元素也如此，类似于将所指向资源嵌入当前标签内。这也是为什么将js脚本放在底部而不是头部。

#### 简述同步和异步的区别

同步是阻塞模式，异步是非阻塞模式。

同步就是指一个进程在执行某个请求的时候，若该请求需要一段时间才能返回信息，那么这个进程将会一直等待下去，直到收到返回信息才继续执行下去；

异步是指进程不需要一直等下去，而是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行的效率。

#### 请说出三种减低页面加载时间的方法

1. 压缩css、js文件
2. 合并js、css文件，减少http请求
3. 外部js、css文件放在最底下
4. 减少dom操作，尽可能用变量替代不必要的dom操作

#### 前端开发中，如何优化图像？图像格式的区别？

1. 不用图片，尽量用css3代替。 比如说要实现修饰效果，如半透明、边框、圆角、阴影、渐变等，在当前主流浏览器中都可以用CSS达成。
2. 使用矢量图SVG替代位图。对于绝大多数图案、图标等，矢量图更小，且可缩放而无需生成多套图。现在主流浏览器都支持SVG了，所以可放心使用！
3. 使用恰当的图片格式。我们常见的图片格式有JPEG、GIF、PNG。
   1. 基本上，内容图片多为照片之类的，适用于JPEG。
   2. 而修饰图片通常更适合用无损压缩的PNG。
   3. GIF基本上除了GIF动画外不要使用。且动画的话，也更建议用video元素和视频格式，或用SVG动画取代。
4. 按照HTTP协议设置合理的缓存。
5. 使用字体图标webfont、CSS Sprites等。
6. 用CSS或JavaScript实现预加载。
7. WebP图片格式能给前端带来的优化。WebP支持无损、有损压缩，动态、静态图片，压缩比率优于GIF、JPEG、JPEG2000、PG等格式，非常适合用于网络等图片传输。

#### 浏览器是如何渲染页面的？

1. 解析HTML文件，创建DOM树。自上而下，遇到任何样式（link、style）与脚本（script）都会阻塞（外部样式不阻塞后续外部脚本的加载）。
2. 解析CSS。优先级：浏览器默认设置<用户设置<外部样式<内联样式<HTML中的style样式；
3. 将CSS与DOM合并，构建渲染树（Render Tree）
4. 布局和绘制，重绘（repaint）和重排（reflow）

#### 一次js请求一般情况下有哪些地方会有缓存处理？

1. 浏览器端存储
2. 浏览器端文件缓存
3. HTTP缓存304
4. 服务器端文件类型缓存
5. 表现层&DOM缓存

#### 谈谈以前端角度出发做好SEO需要考虑什么？

1. 了解搜索引擎如何抓取网页和如何索引网页
2. meta标签优化
3. 关键词分析
4. 付费给搜索引擎
5. 链接交换和链接广泛度（Link Popularity）
6. 合理的标签使用

#### 请写出几个html5新增的标签，并说明其语义和应用场景

section：定义文档中的一个章节

nav：定义只包含导航链接的章节

header：定义页面或章节的头部。它经常包含 logo、页面标题和导航性的目录。

footer：定义页面或章节的尾部。它经常包含版权信息、法律信息链接和反馈建议用的地址。

aside：定义和页面内容关联度较低的内容——如果被删除，剩下的内容仍然很合理。

#### iframe的优缺点？

优点：

a. 解决加载缓慢的第三方内容如图标和广告等的加载问题

b. iframe无刷新文件上传

c. iframe跨域通信

缺点：

a. iframe会阻塞主页面的Onload事件

b. 无法被一些搜索引擎索引到

c. 页面会增加服务器的http请求

d. 会产生很多页面，不容易管理。

#### 对前端界面工程师这个职位是怎么样理解的？

a. 前端是最贴近用户的程序员，前端的能力就是能让产品从 90分进化到 100 分，甚至更好

b. 参与项目，快速高质量完成实现效果图，精确到1px；

c. 与团队成员，UI设计，产品经理的沟通；

d. 做好的页面结构，页面重构和用户体验；

e. 处理hack，兼容、写出优美的代码格式；

#### 对于MVVM的理解？

MVVM 是 Model-View-ViewModel 的缩写。  
Model代表数据模型，也可以在Model中定义数据修改和操作的业务逻辑。  
View 代表UI 组件，它负责将数据模型转化成UI 展现出来。  
ViewModel 监听模型数据的改变和控制视图行为、处理用户交互，简单理解就是一个同步View 和 Model的对象，连接Model和View。  
在MVVM架构下，View 和 Model 之间并没有直接的联系，而是通过ViewModel进行交互，Model 和 ViewModel 之间的交互是双向的， 因此View 数据的变化会同步到Model中，而Model 数据的变化也会立即反应到View 上。

#### 什么是vue生命周期及其作用？

Vue 实例从创建到销毁的过程，就是生命周期。从开始创建、初始化数据、编译模板、挂载Dom→渲染、更新→渲染、销毁等一系列过程，称之为 Vue 的生命周期。

它的生命周期中有多个事件钩子，让我们在控制整个Vue实例的过程时更容易形成好的逻辑。

#### Vue实现数据双向绑定的原理

vue实现数据双向绑定主要是：采用数据劫持结合发布者-订阅者模式的方式，通过Object.defineProperty（）来劫持各个属性的setter，getter，在数据变动时发布消息给订阅者，触发相应监听回调。当把一个普通 Javascript 对象传给 Vue 实例来作为它的 data 选项时，Vue 将遍历它的属性，用 Object.defineProperty 将它们转为 getter/setter。用户看不到 getter/setter，但是在内部它们让 Vue 追踪依赖，在属性被访问和修改时通知变化。

vue的数据双向绑定 将MVVM作为数据绑定的入口，整合Observer，Compile和Watcher三者，通过Observer来监听自己的model的数据变化，通过Compile来解析编译模板指令（vue中是用来解析 {{}}），最终利用watcher搭起observer和Compile之间的通信桥梁，达到数据变化 —>视图更新；视图交互变化（input）—>数据model变更双向绑定效果。

#### Vue组件间的参数传递

1.父组件与子组件传值  
父组件传给子组件：子组件通过props方法接受数据;  
子组件传给父组件：$emit方法传递参数

2.非父子组件间的数据传递，兄弟组件传值  
eventBus，就是创建一个事件中心，相当于中转站，可以用它来传递事件和接收事件。项目比较小时，用这个比较合适。（虽然也有不少人推荐直接用VUEX，具体来说看需求咯。技术只是手段，目的达到才是王道。）

#### vue路由的钩子函数

首页可以控制导航跳转，beforeEach，afterEach等，一般用于页面title的修改。一些需要登录才能调整页面的重定向功能。

beforeEach主要有3个参数to，from，next：

to：route即将进入的目标路由对象，

from：route当前导航正要离开的路由

next：function一定要调用该方法resolve这个钩子。执行效果依赖next方法的调用参数。可以控制网页的跳转。

#### 对keep-alive 的了解？

keep-alive是 Vue 内置的一个组件，可以使被包含的组件保留状态，或避免重新渲染。  
在vue 2.1.0 版本之后，keep-alive新加入了两个属性: include(包含的组件缓存) 与 exclude(排除的组件不缓存，优先级大于include) 。

#### 什么是vue的计算属性？

在模板中放入太多的逻辑会让模板过重且难以维护，在需要对数据进行复杂处理，且可能多次使用的情况下，尽量采取计算属性的方式。好处：①使得数据处理结构清晰；②依赖于数据，数据更新，处理结果自动更新；③计算属性内部this指向vm实例；④在template调用时，直接写计算属性名即可；⑤常用的是getter方法，获取数据，也可以使用set方法改变数据；⑥相较于methods，不管依赖的数据变不变，methods都会重新计算，但是依赖数据不变的时候computed从缓存中获取，不会重新计算。

# .NET基础部分

#### C#中的垃圾回收机制。(了解)

GC就是[垃圾回收](https://www.baidu.com/s?wd=åå¾åæ¶&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YYrjnkmhcznWRvPHTYnvR10ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHTzPWf1n16v)器,一般来说系统会自动检测不会使用的对象或变量进行内存的释放,不需要手动调用,用Collect()就是强制进行[垃圾回收](https://www.baidu.com/s?wd=åå¾åæ¶&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YYrjnkmhcznWRvPHTYnvR10ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHTzPWf1n16v),使内存得到及时的释放,让程序效率更高。

#### 面向对象的主要思想包括什么？（必会）

1）封装：用抽象的数据类型将数据和基于数据的操作封装在一起，数据被保护在抽象数据类型内部，程序的其他部分只有通过被授权的操作（成员方法）才能对数据进行操作。。

2）继承：继承可以解决代码复用问题，子类可以沿用父类方法和属性的一种方式。

3）多态：继承同一父类的不同子类，对父类方法的不同实现。分为方法的隐藏和重写。

#### C#中的接口和抽象类有什么异同？（必会）

**从形态层面上：**

**相同点：**抽象类和接口的都是抽象的，不能实例化，需要子类实现其抽象成员。

**不同点：**

1）抽象类是类的部分实现，可以包含已实现的成员，而接口不能包含任何成员的实现。

2）抽象类与普通类一样，可以继承类和实现接口，而接口只能继承接口，不能实现接口。

**从设计层面上：**

**不同点：**

1）抽象类是对一种事物的抽象，即对类抽象，而接口是对行为的抽象。抽象类是对整个类整体进行抽象，包括属性、行为，但是接口却是对类局部（行为）进行抽象。

2）抽象类作为很多子类的父类，它是一种模板式设计。而接口是一种行为规范，它是一种辐射式设计。

#### 什么是装箱和拆箱？（熟悉）

装箱就是把值类型到引用类型的转换

拆箱就是引用类型到值类型的转换

#### 简述 private、protected、internal 、public 修饰符的访问权限。（必会）

private : 私有成员, 在类的内部才可以访问。

protected : 保护成员，该类内部和继承类中可以访问。

internal: 在同一程序集内可以访问。

public : 公共成员，完全公开，没有访问限制。

#### C#中的委托是什么？事件是不是一种委托？（熟悉）

C#中的委托是一种引用类型，表示具有相同方法签名的方法的引用。

事件不是委托,是委托类型的变量。

#### 线程和进程的区别（了解）

**最根本区别：**进程是操作系统资源分配的基本单位，而线程是CPU调度和执行的基本单位

在开销方面：进程开销较大，线程开销小。

#### 能用foreach遍历访问的对象需要实现 \_\_\_\_\_\_ 接口或声明\_\_\_\_方法的类型。

能用foreach遍历访问的对象需要实现 IEnumerable 接口或声明GetEnumerator 方法的类型**（熟悉）**

#### c#中的三元运算符是什么（熟悉）

C#中有一个三元运算符”?:”，语法为：条件表达式？表达式1：表达式2；

#### 利用operator声明且仅声明了==，有什么错误么?（熟悉）

重载了“==” 就必须重载 “!=”

#### 委托声明的关键字是什么?（熟悉）

delegate

#### 用sealed修饰的类有什么特点？（熟悉）

该类为密封类，不可被继承

#### 在.Net中所有可序列化的类都被标记为什么?（熟悉）

使用Serializable特性标记

#### override与重载的区别（必会）

**Override：方法重写**，**子类对父类中**的方法进行重新实现，要求，父类与子类之间、方法名和参数类型，数量，顺序一致。在C#允许子类对父类的抽象方法或虚方法进行重写，以适应新的需求。

**重载：在同一类中**，存在多个具有相同方法名，而不同参数列表的方法，在调用时根据实际参数确定执行哪个重载方法。

#### 请描述一下ref和out在方法参数传递上的异同点？ref和out（熟悉）

ref和out：

**相同点：**

1）在实参和形参上都必须添加关键字

2）在方法中对参数的设置和改变将会被保留。

**不同点：**

1）ref修饰的参数在方法调用前必须初始化，而out修饰的参数在方法调用时候可以不初始化。

2）在方法中out修饰的参数必须进行赋值，而ref修饰的参数不需要。

3）out修饰的参数主要用于返回值，ref修饰的参数主要用于修改。

#### 请指出.NET中所有类型的基类（必会）

.Net中所有类型的基类是System.Object

#### 在C#中using和new这两个关键字有什么意义，请写出你所知道的意义？（了解）

using 关键字有两个主要用途：

1）作为指令，用于为命名空间创建别名或导入其他命名空间中定义的类型。

2）作为语句，用于定义一个范围，在此范围的末尾将释放对象。

在 C# 中，new 关键字可用作运算符、修饰符或约束。

1）new 运算符：用于创建对象和调用构造函数。这种大家都比较熟悉，没什么好说的了。

2）new 修饰符：在用作修饰符时，new 关键字可以显式隐藏从基类继承的成员。

3）new 约束：用于在泛型声明中约束可能用作类型参数的参数的类型。

#### 接口是否可继承接口?抽象类是否可实现接口?抽象类是否可继承类?（熟悉）

在C#中，接口可以继承接口，但不是实现接口，因为，接口中不能存在实现的成员。

抽象类与普通类无差异，只是不能被实例化而已，因此，抽象可以实现接口，也可以继承类，但抽象类一般作为父类使用。

#### C#中，请简述string str = null与string str =““的区别？（必会）

string str = null 是不分配内存空间,而string str = ““ 给它分配长度为空字符串的内存空间。

#### 概述反射和序列化（了解）

反射就是为了在程序的运行过程中动态的获取类的方法和属性, 优点就是可以实现动态创建对象和编译，体现出很大的灵活性，缺点是对性能有影响。

序列化是将对象状态转换为容易传输的格式的过程。与序列化相对的是反序列化，它将流转换为对象。这两个过程结合起来，可以轻松地存储和传输数据。

#### 如何理解委托（熟悉）

委托是一种特殊的引用类型，允许将方法作为参数进行传递。

委托在使用过程中的一般步骤：类型定义、对象创建和方法绑定，方法调用。

#### 概述三层结构体系？（必会）

表示层(UI)：为用户实现可交互的界面；

业务逻辑层(BLL)：实现业务规则和数据加工；

数据访问层(DAL)：实现与数据存储区的数据交互。

#### 值类型和引用类型的区别？（必会）

C#数据类型分为两大类：值类型是值拷贝和引用类型是地址拷贝。

#### 构造器Constructor是否可以被继承？是否可以被Override?（熟悉）

Constructor不可以被继承，因此不能被重写（Overriding），但可以被重载(Overloading).

#### 是否可以继承String类?（必会）

因为String类是密封类所以不可以继承string类。

#### 什么是受管制的代码（托管代码）？（必会）

unsafe：非托管代码。不经过CLR运行。

#### &和&&的区别（了解）

&是位运算符，表示按位与运算，&&是逻辑运算符，表示逻辑与（and）.

#### 用Singleton如何写设计模式（了解）

单例模式: static属性里面new ,构造函数private

public class Singleton{

//静态私有变量

private static Singleton singleton = new Singleton();

//私有构造方法

private Singleton(){

System.out.println("生成了一个实例！！！");

}

//外部类通过Singleton.getInstance()获取这个类的对象。

public static Singleton getInstance(){

return singleton;

}

}

#### 一列数的规则如下: 1、1、2、3、5、8、13、21、34...... 求第30位数是多少，用递归算法实现。（了解）

　　public class MainClass　{

　　　　public static void Main(){

Console.WriteLine(Foo(30));

}

　　　　public static int Foo(int i) {

　　　　　　if (i <= 0) return 0;

　　　　　　else if(i > 0 && i <= 2) return 1;

　　　　　　else return Foo(i -1) + Foo(i - 2);

　　　　}

}

#### 请简述结构和类的区别（熟悉）

1、类型不同：结构是一种值类型，而类是引用类型。

2、存储不同：结构使用栈存储，而类使用堆存储。栈的空间相对较小.但是访问效率相对较高.堆的空间相对较大.但是访问效率相对较低。

#### 设计模式有哪些？（了解）

1.单例设计模式

2.工厂设计模式

3.代理设计模式

4.观察者设计模式

5.适配器模式

6.原型模式

#### StringBuilder 和 String的区别？（熟悉）

String是不可变化的字符，它只能赋值一次，每一次内容发生改变，都会生成一个新的对象。

StringBuilder是可以变化的字符，每次操作都是对自身对象进行操作，而不是生成新的对象，其所占空间会随着内容的增加而扩充。

# .NET Web开发部分

#### Page\_Onload里面一般会写什么代码？（必会）

通过判断IsPostBack属性判断是否首次加载。在页面首次加载时，编写控件初始化代码和页面变量初始化代码

#### 什么时候会用到cookie,session?它们的区别分别是什么？（必会）

Cookie通过在客户端记录信息确定用户身份，Session通过在服务器端记录信息确定用户身份。

1．session 在服务器端，cookie 在客户端（浏览器）

2．session 默认被存在服务器的一个文件里（不是内存）

3．session 可以放在 文件、数据库、或内存中都可以。

#### 页面之间传值的几种方式？（必会）

**页面间传值：**

1.使用QueryString变量。

2.PostBackUrl()方法。

3.使用Server.Transfer方法。

**应用程序范围内传值：**

1.使用Cookie对象变量。

2.使用Session变量。

3.使用Application对象变量。

#### post和get的区别。（必会）

1. get/post均可向服务器传送页面数据。

2. get一般用来从服务器获取数据，而post一般则用来从客户端向服务端提交更新数据。

3. 数据获取方式：

get，服务器端用Request.QueryString获取变量的值；

post，服务器端用Request.Form获取提交的数据。

4. get传送的**数据量较小**，**不能大于2KB**。**post传送的数据量较大**，一般不受限制

5. get**安全性非常低**，**post安全性较高**。

#### 什么是Code-Behind技术。（熟悉）

代码后置, 即HTML代码与C#代码写在不同的文件中，使得代码更整洁和便于维护.

#### MVC与webform相比有什么优势？（必会）

MVC的优点：

1.通过把项目分成Model、View和Controller，使得复杂项目更加容易维护，减少项目之间的耦合。

2.无需维护视图状态，降低网络带宽占用。

#### webapi与webservice的区别？（熟悉）

答：webapi用的是http协议，webservice用的是soap协议

webapi无状态，相对webservice更轻量级。webapi支持如get,post等http操作

#### MVC有多少种不同类型的结果类型，请详细描述。（熟悉）

有12种，最主要的是ActionResult类，常用的有以下几个对象：

ViewResult  - 给响应流渲染指定的视图；

RedirectResult  - 执行一个HTTP转向到指定的URL；

RedirectToRouteResult  -执行一个HTTP转向到一个URL；

JsonResult  - 序列化一个ViewData对像到JSON格式；

ContentResult  - 写内容到响应流，不需要视图支持；

FileContentResult  - 返回一个文件到客户端；

FileStreamResult  - 返回一个文件到客户端，它提供的是流；

FilePathResult  - 返回一个文件到客户端。

# 数据访问部分

#### DataReader和DataSet的异同（熟悉）

DataReader使用时始终占用Connection对象,它是以游标的方式，每次读取一条数据；

DataSet则是将数据一次性加载在内存中.抛弃数据库连接,读取完毕即放弃数据库连接,比较消耗内存,但是却比DataReader要灵活,可以动态的添加行,列,数据.对数据库进行回传更新操作。

#### ADO.NET中常用的对象有哪些？分别描述一下。（熟悉）

DataSet对象：数据集，表示内存中数据的缓存，可以把它想像成一个临时的数据库。

DataReader对象：读取器，它与DataSet最大的不同是有连接式的，每次对数据库进行存取都会影响到数据库。

Connection对象：连接池，用于连接数据库的对象，表示到数据源的一个唯一的连接。

Command对象：命令对象，表示要对数据库执行的一个SQL语句或一个存储过程。

DataAdapter对象：适配器，该对象是与DataSet配合使用的对象，用于把表填充到DataSet，和更新DataSet等。

# 开发其他部分

#### 编程实现冒泡排序。（熟悉）

int [] array = new int [\*] ;

int temp = 0 ;

for (int i = 0 ; i < array.Length - 1 ; i++) {

for (int j =0 ; j < array.Length-i-1 ; j++) {

if (array[j] > array[j+1]) {

temp = array[j] ;

array[j] = array[j+1] ;

array[j+1] = temp ;

}

}

}

#### 分层式结构究竟其优势何在？（熟悉）

1. 开发人员可以只关注整个结构中的其中某一层；
2. 可以降低层与层之间的依赖；
3. 利于各层逻辑的复用。

概括来说，分层式设计可以达至如下目的：分散关注、松散耦合、逻辑复用、标准定义。