作业内容：

任务1：Nacos与Spring Cloud整合中，AutoServiceRegistration对象如何注入Spring容器的，简述你的理解。

> 1. 导入坐标`spring-cloud-starter-alibaba-nacos-discovery`
> 2. 在`spring-cloud-starter-alibaba-nacos-discovery`中继承了`spring-cloud-commons`
> 3. 在`spring-cloud-commons`中通过spi 自动导入了`AutoServiceRegistrationAutoConfiguration`
> 4. 在`AutoServiceRegistrationAutoConfiguration`中创建`AutoServiceRegistration`的一个实现类的bean对象`NacosAutoServiceRegistration`
> 5. 最后在`NacosAutoServiceRegistration`中构造注入`` `AutoServiceRegistration` ``

任务2：简述NacosDiscovery的注册服务源码流程

> 1. 导入坐标`spring-cloud-starter-alibaba-nacos-discovery`
> 2. 在`spring-cloud-starter-alibaba-nacos-discovery`中通过spi 自动导入了`NacosServiceRegistryAutoConfiguration`
> 3. 在`NacosServiceRegistryAutoConfiguration`中创建了`NacosAutoServiceRegistration` bean 实例
> 4. 而`NacosAutoServiceRegistration`是`AutoServiceRegistration`的实现类，也就是说`AutoServiceRegistration`的bean对象已创建。
> 5. `NacosAutoServiceRegistration`同时继承了`AbstractAutoServiceRegistration`
> 6. 而`AbstractAutoServiceRegistration`实现了`ApplicationListener`接口(该接口的实现方法将会在tomcat启动之后被调用)，并在实现的接口方法中调用了`AbstractAutoServiceRegistration#bind()`方法，然后调用`AbstractAutoServiceRegistration#start()`方法
> 7. 在`AbstractAutoServiceRegistration#start()`中调用了`AbstractAutoServiceRegistration#register()`方法，向注册中心进行注册。
> 8. 注册的时候就会创建一个`NamingService`实例，在`NamingService`实例中调用了`NamingService#registerInstance(...)`方法
> 9. 在`NamingService#registerInstance(...)`方法中向注册中心发送心跳，同时获取服务列表信息。

任务3：什么是Ribbon，什么是OpenFeign，说说你的理解，说说它们的关系?

> - Ribbon
>
>> Ribbon它其实是一个负载均衡的一个组件，它提供多种HTTP 请求的负载均衡策略。
>
> - OpenFeign
>
>> 一种声明式HTTP Rest 客户端，通过注解与接口实现。
>>
>> 能够像调用接口一样进行远程HTTP 调用。
>>
>> 而OpenFeign 中内置了Ribbon组件。