Algorithm

Divide Two Integers

```
class Solution {
public:
    int divide(int dividend, int divisor) {
        int minus = dividend < 0 ? -1 : 1;
        long long dividendTmp = (long long)dividend * minus;
        minus = divisor < 0 ? -1 : 1;
        long long divisorTmp = (long long)divisor* minus;

        if (dividend == INT_MIN)
        {
            if(divisor == -1)
                return INT_MAX;
            if(divisor == 1)
                return INT_MIN;
        }

        int i = 0;
        while(dividendTmp >> i >= divisorTmp)
        {
            ++i;
        }

        long low = pow(2, i - 1);
        long high = pow(2, i);

        long count = high - low;
        while(count > 0)
        {
            long step = count / 2;
            long tmp = high;
            tmp -= step;
            if(divisorTmp * tmp > dividendTmp)
            {
                high = tmp -1;
                count -= (step + 1);
            }
            else
            {
```

```
                    count = step;
                }
            }

            int sign = dividend < 0 ^ divisor < 0 ? -1 : 1;

            return high * sign;
        }
    };
```

这个题虽然标识的是简单难度，但是花了很多时间才算提交成功。主要的困难点在于溢出的处理，以及模拟除法过程。这里用了两个步骤模拟出了除法的效果：

1. 用移位来快速算出结果的上界和下界
2. 在上届和下界之间用二分法搜索出第一个小于等于的所需值（这里参考了c++算法中的lower_bound算法实现）

Review

How to scale Microservices with Message Queues, Spring Boot, and Kubernetes

这篇文章描述了一个典型的网站场景：抢购。当某一时刻有大量的用户请求同时到达时，传统的网络架构会存在着可用性低及可扩展性不佳等缺点。造成这些缺点的一个原因是前端与后端的耦合性太高导致。为了达到对请求“削峰填谷”的作用，即在请求高峰时刻能够有效的缓冲不能及时处理的请求，等待服务器对请求逐步处理，不至于消息无法及时得到处理，造成请求丢弃的情况。这时， 就引入了消息中间件。消息中间件降低了前端和后端的耦合性，既可以缓冲无法及时处理的请求，另外一个好处是可以让后端可以动态的增减，而前端无需关心这种变化，这就可以在业务繁忙时能够动态增加服务器，业务减少时减少服务器。
传统管理服务器需要手动或者通过脚本去开启服务器，配置各种参数。而随着容器技术的发展，各种服务可以通过容器为载体进行发布，方便快捷，且可以利用Kubernetes对容器进行编排，管理，大大减少了运维的负担，可以完成服务器的动态扩容和缩容。
结合目前的学习情况，可以理清楚下来需要学习的东西，spring boot作后台框架，rabbitmq作为消息中间件，docker作为服务的发布容器，kubernetes作为容器的编排工具。这几种技术掌握了就可以对后端开发有一个不错的认识了。

Tech

最近学习Java，就开始尝试利用spring框架搭建web开发环境，刚开始用了spring4进行开发，各种xml配置文件，搞了半天，参数容易出错，磕磕绊绊才搞出了一个demo。后来用上了spring boot，结合initializer，通过几个参数就可以生成一个web环境，十分的方便快捷，因此把这个demo记录下来，后续结合springboot的学习，将会有更加深入的了解。同时后续也可以在这个demo上添加各种组件，正确尽快完成读书群里所说的那个投票程序。
https://github.com/Wangguanyong/SpringBootDemo

Share

最近因为离家独自在外工作，各种焦躁。感觉焦虑感很强，学习东西也很浮躁，虽然看了些书和资料，但是还是感觉大脑空空，没有学习到多少知识，希望后边能够调整好，积少成多，通过做出实际的东西来缓解这种焦虑感。
焦虑感来自于没有成就感。