**①代码存在的BUG：**

可能存在溢出现象，即可能a+b的结果已经溢出，造成数据错误，之后再除以2，并返回答案，最终返回的是个错误答案。

**②看完后，你还自信自己编过的代码没bug、直接应用于社会没有风险吗？**

我不再自信自己的代码没有BUG了，因为很多代码都采用这种简单的处理方法，没有考虑过极端数据的问题。

**③你如何信任团队同伴（其他人）编写的代码？**

其实严格意义上来说，没有任何人的代码值得完全信任的，因为现在没有一种方法可以证明工程中的代码可以做到一定正确。

但是从现实生活中的角度进行分析，如果团队中的同伴和我采用同样的方法来进行代码BUG的避免或者降低，从这个角度上我是可以一定程度上信任的（如果可以由我在代码编写结束后，能够进行一定程度上的测试，则信任程度可以进一步加深）。

**④用什么途径能降低代码风险？**

首先是团队内部针对过去项目以及其他人的一些项目中常出现的错误进行列举，并对各种错误或漏洞提出解决办法，从团队层面上，在代码构建时，尽量避免出现与过去同样的错误。

其次，可以预想一些可能出现的错误，针对这部分错误，提出解决预案，进而预防这部分错误带来的风险。

再次，在每个阶段过程中，在一些关键节点，统一进行代码的风险检测和测试，保证不出问题。同时针对关键代码，进行专业测试，保证不出问题。

最后，在工程结束后，在未正式投入之前，可以进行系统性地、专业性高的代码测试，模拟现实场景，进而降低代码风险。

**⑤如何做好软件产品的质量保证？**

其一，进行一些自动化测试，进而排查一下常见的软件BUG。

其二，由专业测试团队，针对软件实际使用中可能出现的问题，设计一些特殊测试样例和基本测试样例，进行人工测试。

其三，在软件产品的生产过程中，严格按照标准流程进行生产，保证过程中间质量。

其四，在软件产品投入生产之前，做好市场预估，保证产品商业质量。

其五，在正式投入使用之前，内部先进行小规模的试使用，根据用户的反馈及出现的漏洞，及时进行修复和处理。

**⑥感想**

绝大部分程序都只是在限定的条件下，去完成有限的功能。不能将一段代码（即使这段代码在之前的程序中有效或者未遇到过BUG）完完本本、不考虑限制的迁移到另一个工程或程序中，否则可能会造成严重的错误或后果。