1. *Robert Glass does not include all of the quality attributes that are in McCall's list of quality factors. Do you believe the extra attributes in McCall's list are necessary for "quality" software, or is Glass' definition sufficient?*

I believe the Glass' definition is mostly sufficient, but I will only add the maintainability and integrity attributes to it. With maintainability, developers will be able to fix bugs and add new features quickly and easily without causing new bugs. It also allows bringing new developers on board without having them spending much time on trying to understand the code instead of doing actual development. With integrity, the software will be protected from corruption, whether accidental or malicious, that might waste all the time and effort spent on the software. For the rest of the attributes, I think they can exist under one umbrella of one of the Glass' attributes if we added a bit more to their definitions. For example, I think correctness can be a part of reliability, usability part of human engineering, and lastly flexibility, reusability, and interoperability part of portability if we considered program modification, systems coupling, and program reuse as different types of environments.

2. *In Glass' definition, which single attribute do you believe is most important, and why?*

I believe **human engineering** is the most important attribute in Glass' definition. If I, as a user, started to use a software and could not navigate through it or do the basic functionality of it in an easy and straight-forward way, I will immediately start looking for a replacement of such software. That is, poor human engineering reduces productivity and can make it difficult for users to perform tasks correctly and with confidence. Also, by focusing on the human engineering attribute, the software will benefit from the increased user engagement which will accordingly increase users' loyalty and sales profit since users would rather use this software than any other competitor's software because of how user-friendly and easy it's to use your software even if it comes with a higher cost for the same functionality. I think Gmail vs Yahoo! Mail can be a good example for how I would rather use a well-established interface as in Google than an old looking and not so user-friendly interface as in Yahoo! Mail.