Module 1 Assignment

1. Recreate the integer overflow error similar to what happened in the Ariane 5 disaster. In other words, write a few lines of code that demonstrates the problem with attempts to convert a binary 64-bit floating point number (provided by a user, taken as input) to a signed 16-bit (or 32-bit) integer. If you have done this correctly, you should induce an integer overflow error in your code.

2. As you have learned in this course, exceptions require special attention. Exception handling is a strong feature of various programming languages, enabling us to handle errors caused by exceptions. If not handled correctly at runtime, outcomes can be undesirable, and even disastrous (as in the Ariane 5 incident).

Here are some possible ways that the exception handler could handle the overflow error

a) shut down the system

b) print an error message

c) ignore the error and use the value

For each of these, imagine a possible scenario for the Ariane 5 rocket where this would be the wrong way to handle the error. Then describe two better ways to design the exception handler to handle an overflow in the Ariane.

3. The engineers for the Ariane 5 followed the principle of "redundant systems." For every system, there was an identical backup so that if the main system shut down, the identical backup system was there to take over. For example, whenever one screw was needed, the engineers used two so that if one broke, the other screw was there to hold the connection in place. The engineers did this for every critical system on the rocket including the software. Why do you think having a backup software program failed to prevent the crash of the Ariane 5?

4. The software writers for the Ariane 4 used the code from Ariane 4 for the Ariane 5. Because we usually assume API code is correct, the Ariane 5 software writers thought the Ariane 4 routine would work correctly when it was used on the Ariane 5. What should the Ariane 5 software writers have done differently? What should the Ariane 4 software writers have done differently? Give a set of rules that you think everyone (including you) should follow when writing code and when deciding whether to use code that was previously written in order to prevent something like the Ariane 5 crash happening again.

For some inspiration on this question, have a look at the following short papers:

- https://www.cs.jhu.edu/~jorgev/cs106/bug.pdf
- https://homepages.cwi.nl/~storm/teaching/reader/JezequelMeyer97.pdf

5. The subsequent investigation of the Ariane incident ended up holding no one responsible for the failure – as the investigation concluded that the accident was the result of lots of small errors by many different people that all combined to create the disaster. It was not easy to assign blame to any one person. But if nobody is ultimately responsible, how can we prevent such an accident from happening again?

Do you agree that nobody should have been held responsible or do you think that there was one error most responsible for the crash? Suggest a way that a software team can assign responsibility and accountability to prevent lots of small errors creating a major disaster.