### Heathrow Terminal 5 Opening (Cost: Up to $50 million)

The baggage handling software for Heathrow’s new Terminal 5 was tested thoroughly before it opened in March 2008. However, the testing did not take into account real-world usage, which is where the system failed. The result was thousands of bags left behind while the flights left. There were over 500 flights cancelled over the following 10 days of opening.

### Knight Capital Group Stock Trading Bug (Cost: $440 million)

In August 2012, a software bug in Knight Capital Group’s system started trading about 150 stocks erratically, leading to a $440 million loss in half an hour. This is not counting the loss they incurred when their shares went down by 75% in the next two days due to this.

### EDS Child Support System for the UK’s CSA (Cost: $1 billion)

The UK’s Child Support Agency (CSA) started using a very complex software product designed by EDS in 2003. At the same time, there was some internal restructuring. This resulted in two incompatible systems trying to work together.

### Bitcoin Security Flaws (Cost: Over 850,000 bitcoins)

Mt. Gox was the largest bitcoin exchange in the world. In 2011, they lost over 850,000 bitcoins to hackers who took advantage of security flaws in their system.

### Software Bug Heartbleed (Cost: worldwide data leakage)

In 2014, the IT community started discussing a software bug named “Heartbleed” that appeared in the OpenSSL library of TLS protocol. Letting any person connected to the Internet read user information up to 64KB from a computer running OpenSSL process

### Knight almost goes bankrupt (Cost: $440 million)

Knight is one of the key stakeholders of the American stock market, and they almost went bankrupt due to a single software error. Because of the bug, the stock quotations declined by 75%, and the company lost around $440 million. The company settled for amalgamation.

As a software developer I think we should follow some steps to reduce the impact of software failures as well as the cost it take. There are a lots of things we should consider when we develop a software to make it work and do not have the big negative effect when have some bugs. Firstly, we should check the features along with requirements that the software need to meet, and then should take more attention on the important features that can have a big impact when it not working properly. Secondly, we should evaluating alongside with testing the software carefully, especially the functions that take the important role in the software to reduce the negative effect of it when it have some kinds of issues. In addition, before we release it to the customer or to users, so that we will minimize the failures that software can have. Moreover, we should have the backup of software when we develop new version to release. By doing that, we can replace the new version with the working version if the new version has some problems or bugs. And finally, we should ensure that any failures that could occur will not result in personal injury or seriously impact the operation of the system or operational processes by testing carefully and have some alternatives when the software have failures and bugs.

Nguyen Trung Tam - B1910697

### References

<https://raygun.com/blog/costly-software-errors-history/>

<https://www.computerworld.com/article/3412197/top-software-failures-in-recent-history.html>

<https://safebytes.com/10-costly-software-errors-history/>

<http://www.cse.psu.edu/~gxt29/bug/softwarebug.html>

<https://www.isixsigma.com/tools-templates/software/defect-prevention-reducing-costs-and-enhancing-quality/>