SOFTWARE ENGINEERING

DIGITAL ASSIGNMENT-1

Submitted by:

SAI NIKHIL RAGA

17BCI0117

Question-1:

In February 2003 the U.S. Treasury Department mailed 50,000 Social Security checks without a beneficiary name.

a. Find out the reason for this mistake

The main reason for this mistake is that there was a programming error due to which the names were missed.

b. Explain about the activity which they have mentioned as a reason for the mistake

The activity associated with this mistake is Software Testing.

Software Testing:

According to IEEE, Software Testing is an internationally agreed set of standards for software testing that can be used within any software development life cycle and by any organization.

- Testing is necessary and measures software quality. Once faults are removed, software quality and reliability is immediately improved. Software failure can prove to be very expensive and can have a negative effect on a business.
- Test coverage is an important aspect of software engineering that describes the degree to which the code has been tested. It's an arbitrary much contested figure in the software developer community and depends very much on the project and the stipulations set by the software architect. The typical test coverage figure can range from 50% right up to 100% with an average value for a project set to be above 80%.

Benefits of code coverage measurement:

- It creates additional test cases to increase coverage.
- It helps in finding areas of a program not exercised by a set of test cases.
- It helps in determining a quantitative measure of code coverage, which indirectly measures the quality of the application or product

Drawback of code coverage measurement:

- One drawback of code coverage measurement is that it measures coverage of what has been written, i.e. the code itself; it cannot say anything about the software that has *not* been written.
- If a specified function has not been implemented or a function was omitted from the specification, then structure-based techniques can't reveal anything about them and can only look at a structure which is already there.

Question-2:

In October 1999 the $125million NASA Mars Climate Orbiter an interplanetary weather satellite was lost in space due to a data conversion error.

a. Find out the reason for the lost and describe on the testing strategy that helps to avoid the error in critical systems.

Investigators discovered that the software on the spacecraft performed certain calculations in English Units (Yards) where it should have used Metric units (Meters). This is the main reason for the loss of satellite in space.

The testing strategy that helps to avoid the error in critical systems is:

Unit Testing:

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed.
A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Testing Method:

It is performed by using the White Box Testing method.

When is it performed?

Unit Testing is the first level of software testing and is performed prior to Integration Testing.

Who performs it?

It is normally performed by software developers themselves or their peers. In rare cases, it may also be performed by independent software testers.

Testing Benefits:

- Unit testing increases confidence in changing/ maintaining code. If good unit tests are written and if they are run every time any code is changed, we will be able to promptly catch any defects introduced due to the change. Also, if codes are already made less interdependent to make unit testing possible, the unintended impact of changes to any code is less.
- Codes are more reusable. In order to make unit testing possible, codes need to be modular. This means that codes are easier to reuse.

- Development is faster. How? If we do not have unit testing in place, you write your code and perform that fuzzy 'developer test' (You set some breakpoints, fire up the GUI, provide a few inputs that hopefully hit your code and hope that you are all set.) But, if you have unit testing in place, you write the test, write the code and run the test. Writing tests takes time but the time is compensated by the less amount of time it takes to run the tests; You need not fire up the GUI and provide all those inputs. And, of course, unit tests are more reliable than 'developer tests'. Development is faster in the long run too. How? The effort required to find and fix defects found during unit testing is very less in comparison to the effort required to fix defects found during system testing or acceptance testing.
- The cost of fixing a defect detected during unit testing is lesser in comparison to that of defects detected at higher levels. Compare the cost (time, effort, destruction, humiliation) of a defect detected during acceptance testing or when the software is live.
- Debugging is easy. When a test fails, only the latest changes need to be debugged. With testing at higher levels, changes made over the span of several days/weeks/months need to be scanned.

Unit Testing Techniques:

- White Box Testing
- Basis Path Testing
- Testing Module Interfaces
- Testing External Module Interfaces
- Testing local and global data structures
- Testing Error Detection Paths
- Loop Testing

Testing Module Interfaces:

When testing the modules interface, consider the following check list for guidance. In the following discussion, a modules parameter refers to how the module declares the data that it expects to be given when it is called.

The following tests could be conducted on this modules interface.

- Do the numbers of parameters declared in the function itself equal the number of arguments passed into it when it is called from a higher-level module? Any half decent compiler would be able to flag this violation as an error at compile time.

- Do the types of the arguments match the corresponding types of the parameters or can they be automatically converted by the compiler without loss of accuracy of precision. For example, '2.0' above represents a 'double' in 'C', while the function 'volume ()' expects a 'float'.

- Do parameter and argument units match? For example, is the function expecting say, a velocity value measured in feet/sec or meters/sec and does the calling function know which? In other words, check for a miss-match. a function just gets given a number, it doesn't know if it represents Inches, or Millimeters, but you will notice when it crashes!

- Do the order of arguments passed to a function match the order of parameters declared in that function? For example, when calculating the volume of the cylinder has the function been written to expect the height to be the first argument followed by the radius, or vice-versa.