UNIVERSITY OF ZAMBIA

SCHOOL OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Name: Chisomo Y Banja

Student ID: 2021442012

Course Code: CSC 4631

Course: Software Testing and Maintenance

Lecturer : Mr Sinkala

1. From Lehman's Laws, which law do you think is the most relevant today? Why?
   I think Lehman's 1st Law of Software Evolution is still the most relevant in today's world. The law states that a programme that is used in a real world environment must change or become progressively less useful in that environment. This law highlights the fact that software is never static and that it must constantly evolve inorder to meet the ever changing needs, technologies and user expectations. In such a fast pasted, technology driven age, it is imperative that software continually adapts to stay competitive and functional. Software needs constant updates, bug fixes, security updates and addition of features and this illustrates why this law holds critical importance especially in an era of rapid technological advancements.

2. Why is software maintenance considered more expensive than development? Provide examples from the readings.
   Software maintenance costs about 90% more than the cost of initial development. Software maintenance is often more expensive than initial development because of the continuous costs associated with the correction of bugs and updating software to name a few. According to Grubb & Takang (2013), As systems grow and more modifications are made, they become harder to maintain. This very often resuls in higher costs for maintenance as the software ages. According to the "Why Are Software Maintenance Costs So High?" article, unforeseen issues, long term system fixes and compatibility problems contribute to escalating maintenance costs. An example of this is how early software development decisions such as a lack of foresight in architecture can lead to a much higher cost when it comes to later maintenance and updates. This is true to the pareto's principle mentioned in IT Project Management.

3. If a company stops maintaining its software, what are the likely consequences? Give a real-world example.
   If a company were to stop maintaining it's software, it would be at risk of losing customer satisfaction, encountering security vulnerabilities and potentially facing critical system failures. For example, legacy systems that are not updates can become incompatible with newer technologies this leading to operational disruptions. Similarly, failure to maintain software can result in software that gradually becomes unusable and expensive to maintain if you ever do decide to. Another example could be a banking system that fails to update security protocols thus leading to data breaches as attackers will always continue to come up with new methods to explore vulnerabilities.

4. After reading about Git, what are the advantages of using Git over other version control systems?
   The distributed model that Git uses allows each user to have a full copy of the repository, this allows you to work offline and reduce reliance on a central server. Git's branching capabilities also allow a more flexible approach to managing the work of multiple developers without affecting the main codebase. The ability to merge branches and manage changes though a staging area provides control over the development process. These features make Git a very

efficient tool for modern software development compared to older version control systems like CVS which are less flexible and more centralized.