Name: Tarikul Islam Nishat

ID: 21-44632-1

Sec: F

Software Engineering

Lab 1

Brainstorming Practice

1. Provide at least five additional examples of how the law of unintended consequences applies to computer software.

Ans:

The law of unintended consequences:

The law states that "while observing the problem in a deep way, it seems to be clear, and a solution also seems too sure-fire. But at the same time the solution also creates another problem". Examples

Of the law of unintended consequences:

1. Boy and girl internet friendship	In order to facilitate quick, real-time communication, instant messengers were created. The issue with this kind of software is that it allows complete strangers to communicate. With one another.
2. Way of chatting on mobile phone:	Communication through mobile phones has made obstacles to communication obsolete. However, the same technology also revealed a way for nefarious characters to wreak havoc.
3. War (Web – apps span a wide array of applications):	Since computers are now employed in every industry, user tactics have changed. One illustration is a digital dashboard for a car that displays a picture in high-definition resolution. It can be little more than a set of lined hypertext files that present information using text and minimal graphics, but as e-commerce and B2B applications gain importance, web apps are evolving in sophisticated computing environments. This type of image was harmful to create an impact on social network users.

4. Enjoying the videos in piracy DVD/media:	The invention of laser disc technology allowed humanity too easily and inexpensively store data. The audio and video content is being pirated because it is so inexpensive.
Missile tracking system:	It is a noteworthy accomplishment for embedded systems. The Missile tracking system was used to learn about a missile's trajectory. In reality, a software system was created with the intention of determining or tracking a missile's velocity or path. But now this is a curse. Instead of being used for good, this technology is increasingly frequently employed for darker purposes.

2. **Provide a number of examples (both positive and negative) that indicate the impact of software on our society.**

Ans:

Positive	**Negative**
Software is part of our lives; it is in every fiber of our society. Whether it's communication, medical care, transportation, shopping, transactions, or general maintenance, the software is everywhere. Take for example banking these days. We can access our bank accounts online on our laptops or tablet or cellphone using the software. The bank tellers and employees can manage everything on computers using software, including interest calculation, a record of daily transactions, customer requests, etc. Another example is shopping. We can buy everything online, without stepping out of our homes. We can select any item, pay online, get it delivered, and return or cancel it- all using the software.	But apart from positive impacts, there are certain negative aspects of the software as well. Foremost among them is the software that can interfere with the normal operation of our computers or network. Such malicious software can not only steal our data, but it can also hack the network and spread rumors. Similar is the scenario with the use of encryption software to exchange coded data. This can be a beneficial thing in business and a harmful activity if certain people use it for nefarious means (terrorist activities, etc.

3. **Many modern applications change frequently—before they are presented to the end-user and then after the first version has been put into use. Suggest a few ways to build software to stop deterioration due to change.**

Ans:

Create software to prevent damage brought on by change

Instead of creating the software from start, "developing the software involves accommodating the modifications"

It is simple to add on to yet challenging to change.

Many current programs undergo regular changes. The first version is referred to as the version before it is shown to the end user.

The following factors must be taken into consideration by developers in order to prevent bad quality (deter) in a modification of developed software:

1. It must be ensured that any modifications made to one area of the program do not have a negative impact on another area of the program.

2. Software must be modified to accommodate new computer environments or technological advancements.

3. Additionally, it must be remembered that the program shouldn't be dependent on outside components or setups that could evolve over time.

4. Test cases and results are archived and accessible while testing an application. As a result, after modifications, the software can be evaluated again.

5. First, it's important to take the time to comprehend the needs and preferences of the customer.

6. No necessity, no matter how small, should be disregarded.

4. **As software becomes more pervasive, risks to the public (due to faulty programs) become an increasingly significant concern. Develop a doomsday but realistic scenario in which the failure of a computer program could do great harm (either economic or human)**.

Ans:

Like any difficult problem-solving endeavor, software development requires time and focus to be successful. Unfortunately, time is often valued according to economic considerations, and the requirement to deliver finished good results within a completion deadline. Since programs are dependent on the current technology paradigms, commercially accessible resources, and libraries, consumer use of computer programs appears to be declining. True language advancement is relocating to academic institutions and major corporations with intellectual property interests. Between the technological side of computer use and the user and production side, there is a clear and growing divide.

AI is the newest area of discussion on this subject, and there is a lot of worry about AI and its potential for "rogue" AIs to take over systems or cause considerable harm. The majority of these worries stem from a lack of knowledge of fundamental human reasoning and intellect, with the other portion coming from attempts to employ conventional computer programming techniques to produce trustworthy applications or Apps. There is no method to combine possibility and determinism, thus this is really a problem of human programmers not wanting to make that decision. If you make decisions based on probabilities, there are no assurances and you are just playing the odds. If you employ a deterministic approach to decision-making, you will only receive the solution to your issue and will eliminate all other possibilities. Humans should not have unreasonable expectations of AI in both situations.

In response to the op-ed question, placing a current AI system in charge of another crucial system and expecting it to perform above its capabilities is the most likely method a doomsday scenario will be generated. With the way things are going, it won't make any difference whether a person is in charge of the same system or not. Unknown factors that will have catastrophic effects that cannot be anticipated or avoided pose a genuine hazard. Unfortunately, regardless of intention or attention, these situations are typically discovered through trial and error. In the end, no program, clever or not, can perform a task that a human cannot; they can only perform it more quickly. People need to understand that they are simply a reflection of their own abilities and lack of abilities.