[**Homework 1**](https://github.com/hendraanggrian/IIT-ITM511/blob/assets/assignments/hw1.pdf)**: Introduction**

**Problem 1**

*Explain why professional software that is developed for a customer is not simply the programs that have been developed and delivered.*

It is easy to perceive writing lines of code as the only responsibility of a professional software developer. However, this limited point of view disregards the fact that there are extensive processes before coding, such as specifying customer requirements, comprehensive product design, maintaining proper documentation and assuring quality through software testing. Even after delivering the product, the engineering team must oversee critical security patches, user technical support and a roadmap for future updates. Therefore, professional software development is a perpetual cycle of improvement and collaboration that requires as much expertise as the source code itself.

**Problem 2**

*What is the most important difference between generic software product development and custom software development? What might this mean in practice for users of generic software products?*

The most differentiating point of generic and custom software development is the intended user the product is supposed to serve. Generic software products are designed with a larger target audience in mind, they are purchasable by any customer whereas custom software products are geared toward specific customers. In the context of generic product users, they are often cheaper due to mass marketing, have more feedback from a larger audience, and readily available as opposed to specialized software.

**Problem 3**

*Briefly discuss why it is usually cheaper in the long run to use software engineering methods and techniques for software systems.*

By adhering to the standards of software engineering techniques, we could prevent misconceptions and known bugs even though other solutions might exist. The existing software engineering methods are the result of testing and utilization over many years. In my opinion, it is the familiarity with the structured process and the calculated risk that will lead to financial benefit.

**Problem 4**

*Software engineering is not only concerned with issues like system heterogeneity, business and social change, trust, and security, but also with ethical issues affecting the domain. Give some examples of ethical issues that have an impact on the software engineering domain.*

Some examples of ethical concerns in the software engineering space are:

1. **Differing perspectives:** One common instance of an ethical issue in the professional software development industry is a disagreement with the management, be it the policies they enact or the overall project they are working on.
2. **Deferred responsibilities:** In another example, a company may deliberately decide to drop a critical testing plan to save costs or time, leaving the employee with an ethical dilemma to report the misconduct.
3. **Privacy intrusion:** It can also be a privacy issue such as collecting sensitive customer information without permission.

These moral questions are of concern in software engineering because even though they are not inherently technical, they still have a profound impact on how we create software products.

**Problem 5**

*Based on your own knowledge of some of the application types discussed in****Section 1.1.2****, explain, with examples, why different application types require specialized software engineering techniques to support their design and development.*

Software products come in many forms with each solving different problems and having a unique set of challenges. For example, user interface and experience are the cornerstones of entertainment systems for digital media consumption, whereas they are not as important in batch processing systems where the work is mostly invisible to the end users. Systems of systems deal with tight integration between software products, this level of integration is less apparent in stand-alone applications for the obvious reason: they are supposed to work alone (Sommerville, 2016, p. 25). The diversity in software applications ensures that distinct engineering techniques are applied. However, all applications must still adhere to the basic principles of software engineering such as understanding the Software Development Life Cycle (SDLC), expecting good performance, managing requirements and reusing existing solutions.

**Problem 6**

*Explain why the fundamental software engineering principles of process, dependability, requirements management, and reuse are relevant to all types of software systems.*

Software engineering fundamentals are relevant because it is beneficial to apply them regardless of the application type. Being familiar with the software development process will result in efficient time management. On the other hand, building reliable software ensures that the application can withstand difficult circumstances like hardware failures, loss of network connection and other dependability issues. Requirements define the application's objective based on the customer's specifications. Finally, repurposing existing resources is generally encouraged to cut costs and improve overall quality, since existing components are already extensively tested (Sommerville, 2016, p. 26).

**Problem 7**

*Explain how electronic connectivity between various development teams can support software engineering activities.*

Software development is an increasingly complex task (particularly in large systems) that requires a group of collaborating engineers. Electronic connectivity – usually in the form of online communication – facilitates the balancing of workload between participating software engineers by enabling them to operate in a separate location (or time zone). This resource sharing is not limited to the source code, as design blueprints and other documentation could help in creating a centralized knowledge base. Moreover, constant engagement would foster team intimacy and promote a productive environment.

# Bibliography

Sommerville, I. (2016). Software Engineering. In *Software engineering diversity* (10 ed., pp. 25-26). Pearson Education.