**INTRODUCTION TO SOFTWARE ENGINEERING.**

1. Software engineering is a branch of science that deals with the design, development, testing and maintenance of software applications. It involves the use of engineering principles to build software based on structured methods.

It is different from traditional programming which primarily involves writing codes and follows a less formal approach.

1. Software Development Lifecycle, SDLC, has the following phases:

a. **Design**: involves the plan of the software system`s structure and components. Hence, it is here that the software architecture is designed.

b. **Development**: this is where the code for the software is written based on the structure and design of the project.

c. **Testing**: this involves the identification and fixing of bugs to ensure the software meets appropriate standards.

d. **Deployment**: this is the stage where the software is prepared for release.

e. **Maintenance**: this involves fixing any issues after the software application has been released.

1. The agile and waterfall models are two common methods for software development. Agile is repetitive and flexible and so allows for continuous improvement throughout the project`s lifecycle. Waterfall, on the other hand, is a system that requires the completion of each project phase before moving to the next.

Agile is best for projects that need flexibility and customer interaction while waterfall is suitable for projects and a clear end goal.

1. Requirements engineering is the area of software engineering that deals with the process of developing and verifying the system requirements.

It helps to achieve the main goal of making sure that the delivered system meets the customer`s needs, hence it is very important in software development. The process is about defining, documenting and maintaining the requirement for a software system.

1. Modularity deals with the process of breaking down a software system into small components or modules that are easy to work with.

Understanding the concept of modularity makes maintenance easy as tasks such as locating and fixing bugs and addition of new features can be simplified.

It aids scalability as modules can be improved on, tested and deployed independently, making it easy to scale the system.

1. **Unit testing**: this is usually done by developers during the development stage to check the smallest parts of the software system to ensure that they work as intended.

**Integrated testing**: this is performed to check the interaction between co-operating units to ensure that they work well together.

**System testing**:

1. A software project manager performs the roles of planning, scheduling and managing the delivery of software. He/she assembles the team, works with the them and allocates roles to the different members. Project managers are faced with the challenges of insufficient budget, difficult deadlines, inadequate skills amongst others.
2. Software maintenance is the process of updating, modifying and changing software after they have been launched to meet customer needs. Some types of software maintenance are:
3. **Adaptive maintenance**: this deals with making updates to ensure compatibility between changing hardware and software environments.
4. **Corrective maintenance**: this is done to fix bugs and errors in a software system.
5. **Preventive maintenance**: this involves changes that are done to prevent errors from happening in the future.

Maintenance is important because it helps software adapt to changes, fix issues and add new features.

1. Some ethical issues faced by software developers are:
2. Breach of privacy
3. Security issues
4. Theft of intellectual property

Software engineers can make sure they adhere to ethical standards by:

1. Adopting best practices
2. Prioritizing user privacy and data protection
3. Maintaining professionalism at all times

**REFERENCES:**

1. Software engineering, Wikipedia.
2. Geeksforgeeks.com
3. Forbes advisor, www.forbes.com
4. Zibtek.com