Advanced Software Technologies – Въведение – резюме

1. Какво е Software? - инструкции и данни, които управляват работата на компютърните системи. Това е нематериалният еквивалент на физическия хардуер и играе ключова роля за функционирането на сложни системи.

2. Атрибути на качествен Software

- **Reliability**: Работи надеждно при определени условия.
- **Scalability**: Може да се справя с нарастващи натоварвания без влошаване на производителността.
- **Usability**: Лесен и интуитивен за употреба.
- **Maintainability**: Позволява ефективни модификации и актуализации.

3. Software Engineering - структурирани подходи за проектиране, разработка, тестване и поддръжка на софтуерни системи. Целта му е създаването на надежден, ефективен и поддържаем софтуер.

4. Жизнен цикъл на Software включва следните фази: 1/ **Conceptualization 2/Design and Development 3/Testing 4/ Deployment 5/Maintenance**

5. Functional vs. Non-Functional Requirements

- **Functional Requirements**: Определят какво системата трябва да прави (напр. функционалност за вход).
- **Non-Functional Requirements**: Определят качествата на системата като производителност, сигурност и използваемост.

6. Software Architecture - дефинира структурата на софт. система, включително нейните компоненти и взаимодействията между тях.

Осигурява основа за постигане на надеждност, мащабируемост и поддръжка.

7. Testing в Software Development - процеси като unit testing, integration testing и system testing. Помага за ранното откриване и отстраняване на грешки.

8. Ключови процеси в Software Engineering

- **Prototyping**: Създаване на опростена версия на системата за тестване и подобрение.
- **Refactoring**: Подобряване на вътрешната структура на кода без промяна на функционалността.
- **Risk Management**: Идентифициране и смекчаване на потенциални рискове за проекта.

9. Agile Methodology - набляга на гъвкавост, работа в екип и итеративни цикли на разработка за бързо доставяне на функционален софтуер, който се адаптира към променящите се изисквания.

10. Principles of Good Software Architecture

- **Low Coupling**: Намалява зависимостите между компонентите.
- **High Cohesion**: Осигурява, че компонентите изпълняват конкретни, добре дефинирани задачи.
- **Extensibility**: Улеснява добавянето на нови функционалности.
- **Reliability and Maintainability**: Гарантира, че системата е устойчива и лесна за модификация.

11. Continuous Integration - процес, при който промените в кода често се интегрират, тестват автоматично и се внедряват ефективно. Това намалява предизвикателствата при интеграция и осигурява ранно откриване на грешки.

12. Technical Debt -сложност или лоши практики в кода, които затрудняват бъдещи промени. Управлението му е важно за здравето на проекта.