**Software Design Fundamentals Week 1**

* Software is everywhere. Almost everything that makes the world spin uses software
* Software refers to computer programs that provide desired function and performance, data structures that manipulate the information and documents that describe the operation and use of the program
* Application Software
  + Handles user request
* System Software
  + Handles system requests
  + OS, Utility software,
  + Is developed or engineered, not manufactured
  + Software doesn’t wear out
  + Software continues to be custom built
* Types of Software
  + System Software
  + Application Software
  + Engineering/Scientific Software
  + Embedded Software
  + WebApps
* Software Crises
  + Chronic Affliction
  + Can’t estimate cost, time, manpower
  + Major Symptoms of Software Crises
    - Over budget
    - Schedule Slippage
    - Poor Quality
  + Major Causes
    - Communication, not understanding properly
* Project Success rate
  + Smaller Projects have a higher chance of success than bigger projects
* Reel suggests a 5 point common sense approach:
  + Start on the right foot
  + Maintain momentum
  + Track progress
  + Make Smart Decisions
  + Conduct Post-mortem Analysis
* Software development Cycle
  + 1. Planning
    - Economical Feasibility
    - Schedule Feasibility
    - Technical Feasibility
    - Operational Feasibility
    - Feasibility Report
    - Scope of the Project
  + 2. Analysis
    - Current System Analysis
    - Proposed system Analysis
    - Requirement Elicitation techniques
  + 3. Design
    - System Context Diagram
    - Data Flow Diagram
    - ER Diagram
    - Sequence Diagram
    - Use case Diagram
  + 4. Implementation (Coding)
    - Implement design using any tool
  + 5. Testing a
    - Unit testing
    - Component Testing
    - Module testing
      * Testing Department will do: Black Box testing, White Box testing
    - SQA: Software Quality Assurance department
  + Integration( Deployment):
  + 6. Maintenance
    - Corrective maintenance
    - Adaptive maintenance
    - Enhancement Maintenance
    - Preventive maintenance
  + Models
    - LSM, Prototyping, RAD, Incremental,
    - Agile development: Scrum, XP
* Well Designed Apps
  + Complete within timeline, budget, and performance criteria
  + Customer-Friendly: Great software always does what the customer wants
  + OOP programmer: Great software is object oriented and easy to extend
  + Design-guru programmer: Great software is when you use tried and true design patterns and principles
* Project decomposition/Project Partitioning/Project Modularity
  + Splitting project into modules efficiently and reasonably
  + Divided based on functionality
  + Too many modules means long implementation time
* Great software in 3 steps
  + Make sure the software does what the customer wants
  + Apply basic object-oriented principles to add flexibility
  + Srive for a maintainable and reusable design
* Array
  + A container
  + Each element is stored in a specific index of the container
  + Used to store multiple values in a single variable
  + Define the variable type with brackets: String[] cars;
  + String[] cars = {“Volvo”, “BMW”}
  + Index starts at 0
* Multidimensional Array
  + Int[][] nums = { {1, 2, 3 ,4}, {5, 6, 7} }
  + Array inside array
* Class Diagram
  + Name of class = Student
  + Attributes = StudentId, Name, Last name
  + Methods = Study(), Revise()
  + All methods are public
  + All attributes are private
* Primary key
  + Phone number cant be primary key, although it is unique, but could be null.
  + Must be unique and not null

A screenshot of a computer

Description automatically generated