Lecture #1 5-9

SCD

Software Construction

Implementation < Logic Building

- **SDLC**

① Requirement ② Design ③ Implementation
④ Testing ⑤ Deployment ⑥ Maintainance

L#1 **AI** 5-9

→ **What is Intelligence**

→ **What is AI?**

- Human like
- Create machines that can think and act like human
- ML: train the machine to make the decisions.
- **Rational - Like**
- Make decisions according to situation
- Human Like decission making policy

- **Reflexive Action**
  - Actions that are performed suddenly on a special condition

    eg: Touching a very hot thing you immediately pull off your hand.

    eye blinking
- **History of AI**
  - 1960s → start
- **AI Applications**
  - Robotics

Lecture #3 12-9-2023

SCD

Software Construction

① Software Engineering → Design, Documentation
② Coding

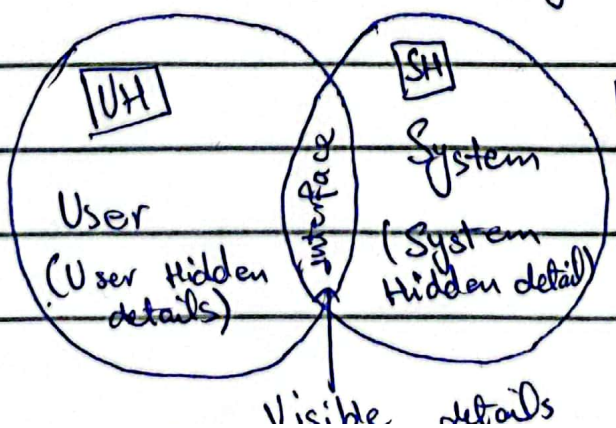
→ # Nature of Problem

Functional Requirement

① Understand Problem Domain
② Requirement Gathering (What is required?)
   ⓐ User requirement (User language) what?
   ⓑ System requirement How? (How to implement)

Non-functional Requirement

① Product Requirement (Hardware Req. # Product Deployment
② Organization (Organization Proof) (Security)
③ External Requirement (All factors that are imported from external system)
↳ Python old version.
↳ Ecomerce purchase payment method failed



WRSPM
W → World (Domain)
R → Requirement
S → Specification
P → Programming
M → Machine

③ Architorecture

→ Decompose an enterprice system into independent sub-system that have value in the system.

→ How these sub system interact

→ Principles and guidance for the design evolution over time.

Lecture #3 AI 12-9-2023

→ Agent

↳ Intelligence is measured by

① Situated in a enviroment

② Autonomous (Independent)

③ Proactive ④ Social (Interactive)

→ Task environment

Types / Structure of agents

1- Simple Reflex Agent → Have no history

2- Model based

3-

4-

Lecture #4 13-09

SCD

→ Module depends on other

Persons or companies may own cars. The car owner ID is the ID either the person or company, that owns the car. A car may have only one owner (person or company). A car may have a loan on multiple loans. A bank provides a loan to a person or a company for the purchase of a car. Only the loan owner may obtain a loan on the car. The car owner type and the loan customer type indicate whether the car owner loan holder is a person or compay.

Make a list of nouns and verbs

classes

1. Person
2. Company
3. Car
4. Loan
5. Bank

Lecture #5 20-9

SCD

Domain Modelling

→ Understand the problem

WRSPM

↓

World (Domain) → Understand your domain

Requirement Elicitation

↳ User requirement (what the user need?)

User → Between user & environment

WR(S)PM

Environment

Software Architecture

→ Software Components (Sub-system) | that have their own value

→ How they interact (sub-system)

→ Independant feature and how they provide facility and interact with each other

SA Model

Module → Dependant (Can be ~~many~~ manay)

① Pipe & filters (Same input, output) (Compiler)

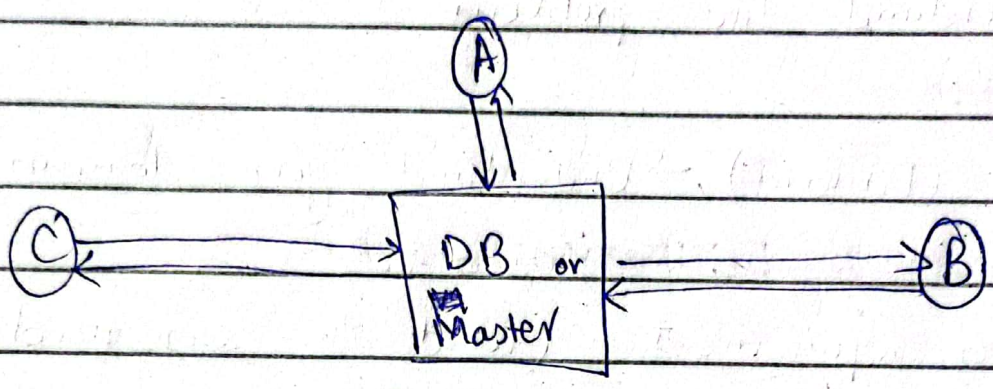
② Blackboard (Central database, Different Modules)

③ Layered (MVC, MVM) (Implement in different layers and don't interact)

④ Client Server (Request, Response or Served)

⑤ Event based (ASP .NET) Output on specific event

→ Architectural Models that can be follow together or sp seperately



Blackboard

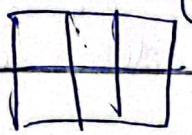
Event based => Occurs on specific event

Software Design

System Structure (Decompose into Components)
Interaction of components (sub-systems)

→ Modular Decomposition → (Software Design)

↓ Design classes



Software Design

→ Design of a specific (particular) module

Modularity

Breaking down of a component and interaction

A module must have these four

things:

① Coupled - Coupling

② Cohesion

③ Information Hiding

④ Data encupsulation

AI

Problem Solving by Searching

Measuring problem-solving performance

→ Completeness: Always returns the ~~optimal~~ solution (Tell if solution exists or not)

→ Optimality: Optimal Solution

→ Time Complexity: How much time it takes

→ Space Complexity: How much space it takes

Branch factor: Maximum no. of nodes

Search strategies

→ No information about the path or the path cost and do not have additional information.

→ Often called blind search

→ e.g Breadth-first, Depth firsth, Uniform cost, Bidirectional search

Frontier

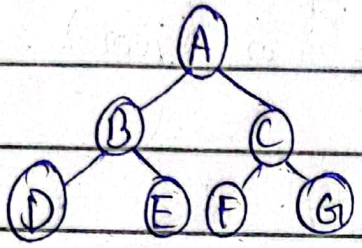
Node that may or may not have child node but they are not expanded or searched.

Breadth - first Search

assumes that every cost is 1

Implementation:

- Fringe is FIFO (Queue)
- Goal = M



Frontier ~~A~~ ~~B~~ ~~C~~ ~~D~~ ~~E~~ ~~F~~ ~~G~~

Expand

Explore: A B C D E F G

-> if the frountier is empty

Solution not Found

is A == Goal

Frontier == Goal

BFS:

① Complete : Yes

② Time : O(b^d) -> exponential in d

③ Space : O(b^d) -> keeps every no in memory (problem)

④ Optimal : Yes (if cost = 1 per step)

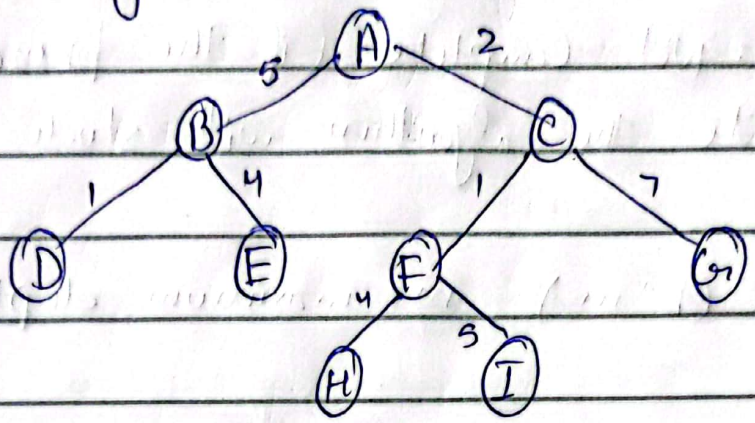
=> Time and Space complexity in exponential is not considered good.

Uniform - cost Search

- If the cost is equal of all nodes then BFS is optimal
- Extention of BFS, depends on costs

Implementation :

- Priority queue
- [X] = g(n)



- Elements are arranged ~~costs~~ according to costs of nodes (Assending order)
- If the cost of 2 nodes are same then we follow BFS (Queue)

UCS:

① Complete: Yes (if b is finite)
② Time: $O(b^{c*})$
③ Space: $O(b^{c*})$
④ Optimal: Yes

Depth First Search

- Always expand one of the deepest level and then back track
- Backtracking search
- Implementation:

LIFO (~~Queue~~) (Stack)

DFS

① Complete : Not Complete (if the branch is infinite the algorithm will stuck in loop)

② Time: $O(b^m)$ m = maximum depth

③ Space:

④ Optimal : does't ~~grentee~~ gurentee best solution

CN

21-9

① Access Link
② Edge devices
③ Network Core

Packets = 2 L

Link Speed = R (bps)

2R

No. of bits in each packet = L

Packet Transmission delay = $\frac{L \text{ (bits)}}{R \text{ (bits/sec)}}$

→ RJ 11 – ethernet

→ RJ 45 – Telephone (landline)

→ Routing Algorithm

→ Store & Forward

→ Arrival > Transmission Rate then this cause queue

BPE

→ Activities (Streamline) $A_1 \rightarrow A_2 \rightarrow A_3$

→ Sequence

Left-to-right | Workflow | → Order

ERP → Enterprise Resource

SCD

25-9

Tightly Coupling: (Not Good) Content coupling

① Module A directly access Module B data member.

② Common coupling

② Module A and B are relied on some global data.

External coupling

③ Module is reling on externally ~~importe~~ imposed format (protocol / interface).

→ XML and Jason

Loosely Coupling: (Highly) Data Coupling

① Module A only pass parameter for requesting functionality of Module B

② Module A sends message to Module B, Message coupling

Medium Coupling: (Somewhat accepted)

① Module A controls the logical flow of module B by passing information or by using flags. (Control Coupling)

② Module A and B rely on some composite data structure changing data structure directly affects the other module. (Data Structure Coupling)

Cohesion:

Weak Cohesion:

. Coincidential Cohesion

① Different parts of module are together just because they are in a file.

Temporal Cohesion

② Different parts/code / functions are activated at the same time.

Procedural Cohesion

③ One part follows the other in time

Communicational Cohesion

④ Similer parts / functions are grouped. They are similer but perform different thing.

Medium

communicational

① All elements operate on same inputs and produces same output.

② One part output serves as input to other part. → Sequential

Strong

① Each operation in module can manipulate object attributes → Object Cohesion
functional

② Every part of the function is necessary for execution of single well-defined function

↓ Functional Cohesion

AI

26-9

→ DFS is better in space complexity then BFS

Space Complexity = bxm

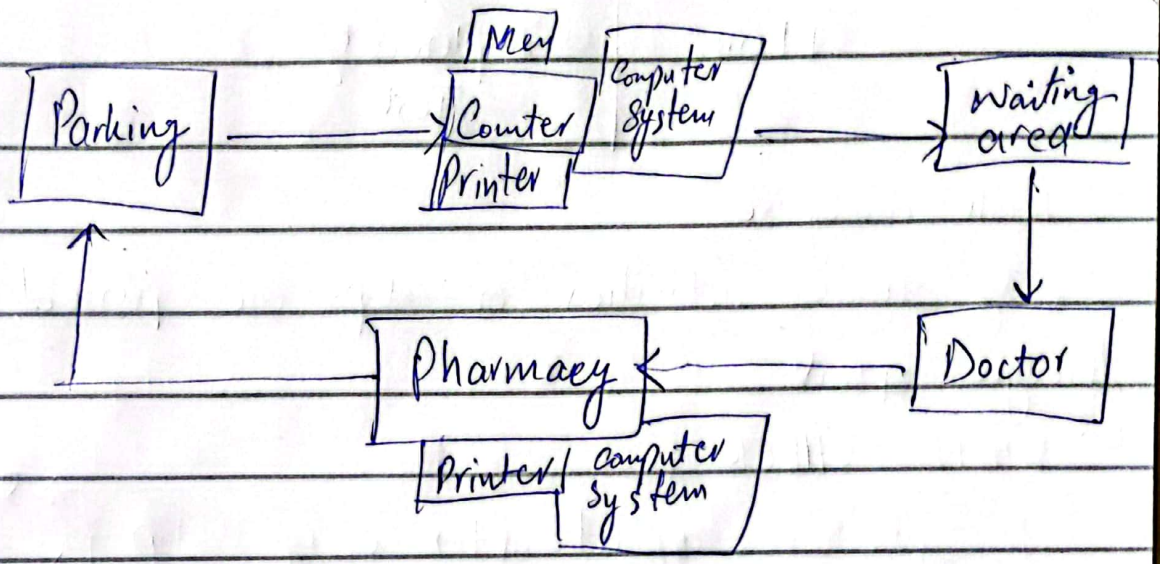
→ Depth Limit Search

↳ The Solution of for the depth limit search, fix the limit of the depth.

→ Termination gurenteed

→ Failure Mode and Effect Analysis

BPE Process



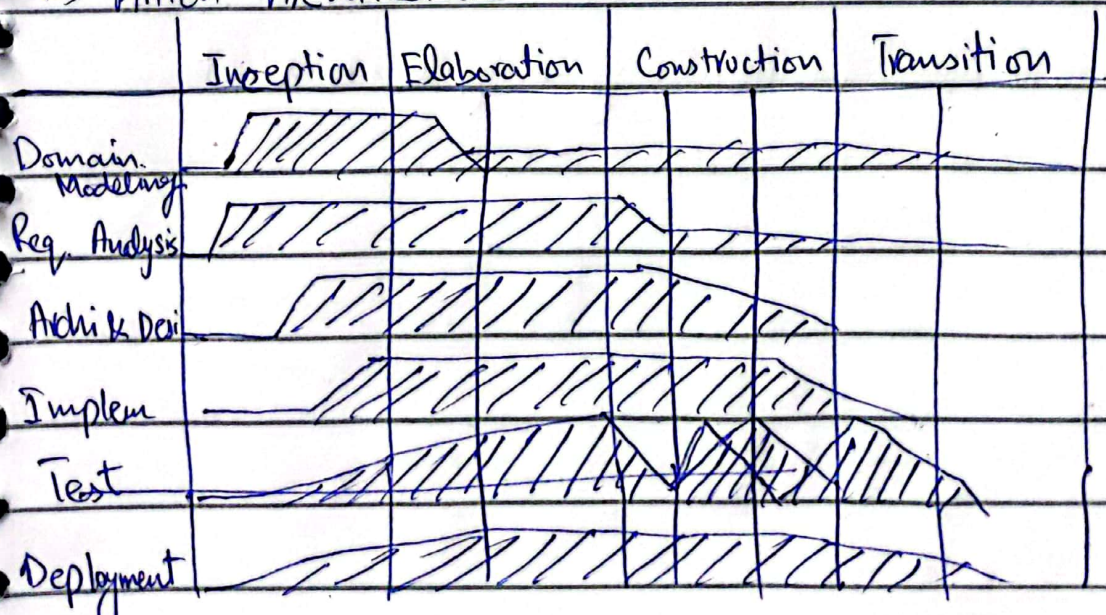
SCD

4-10

- Itrative Model

→ Unified Model

→ ~~Attiot~~ Architecture



→ Understand & Deployment is Difficult

- Spiral Model

Define / Determin Objective	Identify
Plann	Implement / Test

→ Risk analysise

→ Only those activities or steps are passed through spiral

↳ GATE CHECK

→ A gate is applied after every step → Stop / Continue

→ There are multiple way to solve a problem in software development

→ Team is stays motivated

↳ Saves time and Money

- Agile

→ 17 people brainstroming → agile mindset

AI

5-10

• Local Search

→ We use it because it improve a single search iteratively

→ The goal itself is solution

→ Don't keep track or path but keeps a single "current" state

• Hill climbing algorithm

→ Stuck on local maximum

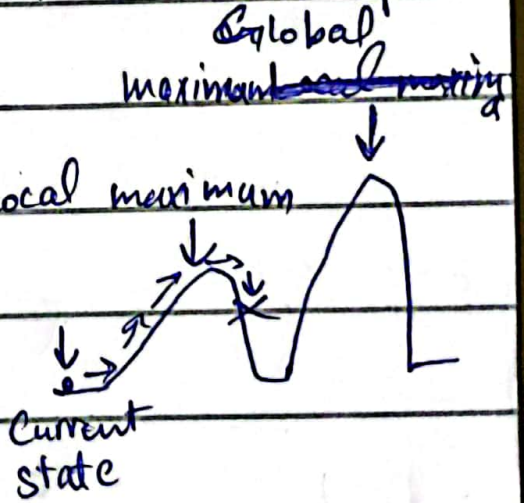
→ Local maximum get worst in higher dimension

→ Local maximum is ~~not~~ one solution but not optimal

→ Continue in same direction (one direction) forward

→ Plateaus

→ Restart or change direction



→ Random restart hill climbing
→ Meta-heuristic
• Optimization Problem: Reduce the Cost →
Local minima, Global maxima
• np Hard Problems

⇒ $f(x_2) > f(x_1)$ → directly move
⇒ $f(x_2) < f(x_1)$ → Then use energy fn.

$$\Delta E = f(x_2) - f(x_1)$$

$$p = \frac{\Delta E}{T}$$ ← assumed number (random)

CN 10-10

Principles Book Pg#133

• P2P • Client Server

No data loss / Reliable

→ Packets => TCP → 100% Packets (Skype)
UDP → (whatsapp video call)

TCP → Reliable → Skype Vide Calling
UDP → Not reliable → Whatapp Calling

SCD

10-10

Agile ~~Principles~~ Values

17 People -> (Senior & Junior)

4 -> Values 12 - Principles

① Individuals and Interactions over Process and Tools

② Working Software over comprehensive documentation

③ Customer Collaboration over Contract negotiation

④ Responding to change

Agile Principles

① Highest priority is to satisfy customer through early and continuous delivery of valuable software.

② Welcome changing requirement ~~are~~ but even late in development. Agile process harness change for the customer is competitive advantage.

③ Deliver working software from couple of weeks to couple of months with the preference to the shortest timescale

④ Business people and developers must work

DSDM
FDD
SCRUM 70%
Crystal
XP
Customs
Lean

together daily throughout the project.

(5) Build the projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

(6) The most efficient and effective method of conveying information to and within a development team is face to face conversation.

(7) Working software is the primary measure of progress.

(8) Agile processes promote sustainable development.

(9) The sponsers, developers and user should be able to maintain a constant pace ~~ident~~ indefinitely.

(10) Continuous attention to technical excellence are good design enhance agility - cost of exploration.

(11) Similarity the art of maximizing the amount of work not done is essential.

The best architectures, requirement and designs emerge from self-~~organizationing~~

teams.

(12) At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

AI 10-10

Adversarial Search

→ Competition

→ Typical AI assumption

→ Zero-sum game (gain of one is lose for other)

Search vs Game

SCD Continue 10-10

IDEA

① Individuals should be able to plan, deliver, monitor and impose the quality and timeliness of their own walk.

② Use data to justify refuge unreasonable demands

Say "Yes" with confidence and "No" with data & options

③ Learn from experience use data from

one piece of work to improve the next

Principles of Personal software Process

Measure stuff (size, time, efforts, defects)

② Measure Consistently

Use orelation to judge usefulness of time to predict future performance

SCD

Self Reading

Team Software Process

PSP

TSP

→ Motivated Team members

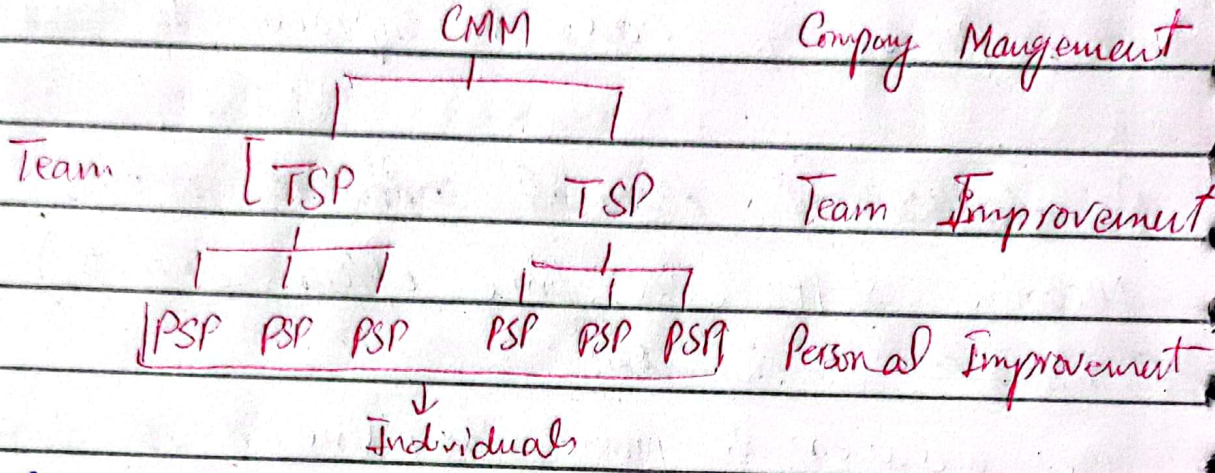
→ Scripts (2-3 page) if there is a prob. then follow this

Humphrey's Idea

- The team should be self-directed
- Definite tasks assign to team individuals
- Communication that plays role in achieving a single.
- Team members are dependant

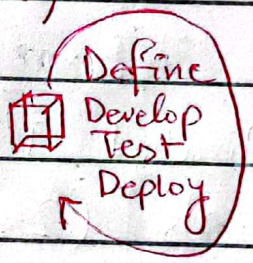
Principles

- Measure the task for each individual/Analy
- Regular meetings
- Rigourous Planning → Planning for next tasks targets (Achieveable goals)



SCRUM

- 1 - 4 week (working software)
- Backlog
- 1 Deployment in Watefall method = 4-6 Deployment in Scrum method



Deployment = Sprint

(Scrum master manage product backlog)

p (Priority) → statement

① Backlog — Simple english statement

② Sprint Planning Meeting

③ Sprint Backlog ← Only for one specific sprint

↳ Daily meeting
- What was previous task
- Problems
- Assignment
- ~~Time~~
- Time Management
- Team Members

④ Finish Product (First Sprint)

⑤ Sprint Review (Problem, issues)

⑤ Conclusion
- conclude the research
- Sum up research thesis

Research Proposal (7-8 pages)
- Summary of research thesis
- Attractive, convice and centered on the ideas that are in research
- Finding => Asumptions
→ Ch 4th and 5th is ~~comb~~ combined
- Citation is must

AI
Minimax Algorithm
→ 2 player game and each plays optimally
→ Best of one is worst of other
Max starts from $-\infty$ $\max(-\infty, 0)$
Min starts from $+\infty$ $\min(+\infty, 0)$
→ Algoritum implementation

Multiplayer games
→ Two-sum cannot be follow
→ Player make alliance

Alpha beta pruning

→ based on min-max but it eliminate exponential space complexity

$$[\overset{\alpha}{-\infty}, \overset{\beta}{+\infty}]$$

→ Prune whenever $\alpha \geq \beta$ for max | $\alpha \leq \beta$ for min

if $\alpha \geq \beta$ or $R \geq 2$ then no need to explore the β branches

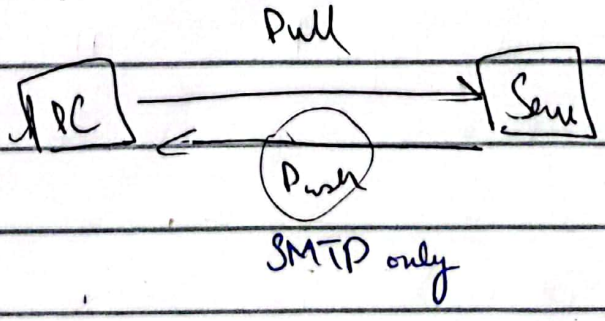
→ Alpha-beta is best for pruning trees and it complexity is $(b^{\frac{m}{2}})$ or $b \hat{} (m/2)$

→ For worst case it is $b \hat{} m$

CN

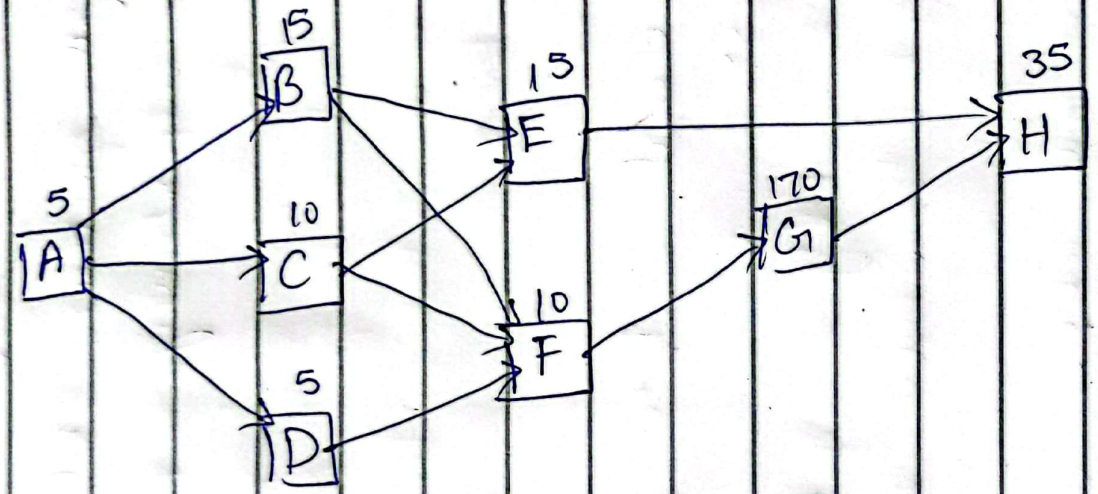
12-10

- Protocol
- TCP vs UDP
- HTTP
- Presistant HTTP and non-Presistant HTTP
- Cookie



Koll business center

Country Engineers Design Department

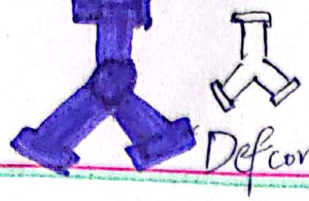


Time ⇒ ABEH = 70

ACEH = 65

ACFGH = ~~225~~ 230

ADFGH = 225



Defcon. 1

Sound Rush

SCD

17-10

Error: Illegal operations that results in abnormal/malfunctioning of a program.

① Syntax
② Logical
③ Runtime

Approach

- Active: Program handle at realtime
- Passive: After inputing wrong that the database cannot store.

Error handling

- Non-functional attributes
- Correctness
- Robustness - ability to handle many ↑ type of inputs that might be correct or incorrect

Technique

- Return a null value
- Substitute with next valid value
- Logs and time stamp
- Error code
- Shutdown

Exception Undesireable situation that can arrive in the program. There are 2 types:

① **Check** - Compiler (Compiler) can recognise
② **Uncheck** - Compiler can't recognise (Runtime Error)

try:
check set of instruction that can cause exception

caught:
handling ~~solution~~ routine for those exceptions if they occur

throw:
new input mismatch exception

throws:
① Class signature ② Multiple

final:
- runs compulsory if the exception occurs or not

AI 17-10

Genetic Algorithm

-> Implement real life genetric operations in AI

-> choose the survival one => better solution

-> Mutation - Swap any random bit

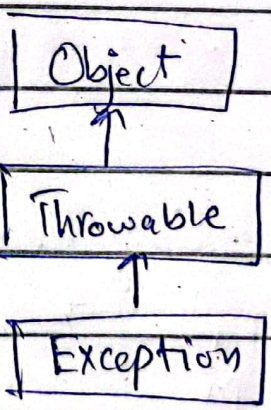
- Evaluation function

SCD 18-10

Fault Tolerance

Collection of techniques that increase software reliability by detecting error and then recovering from if possible or containing their effects of recovery if possible.

1. Backup
2. Retrying
3. Auxilary code (Time tracking of events)
4. Voting Algorithm (Democracy with different scenario)
5. Replacing (errenous input that phony input)
6. Shutdown & Restart



=> Every individual solution is called chromosome

- **Fitness function**
  - Attempt to maximize the function
    ↳ mathmatical function
- **Roulette**
  ↳ Highest fitness value = higher ~~probability~~ probability
  8- Queen Problem
  Fitness -> Selection -> Pairs -> Cross-Over-Mutation

CN 19-10

DNS

AQ: how to map between IP address and name, and vice versa?

AQ: When to we cname, record, alias and URL

Attacking DNS

SCD 24-10

- **Design**
  - Is a sloppy process
  - Is a wicked process
  - Is about trade off, priorities and restriction
  - ~~Is non-deterministic~~

- Is non-deterministic & it is a heuristic process]→ Error and trial (Improve)
- Is emergend

Charateristic

- Minimal complexity
- Easily maintainable
- Loose coupling
- Extensibility
- Reusability
- High ~~form~~ fan-in : a class is used by many other classes
- Low to medium fan-out : a class uses low to medium number of other classes
- Portability
- Learnness
- Stratification (sorted format)
- Keep levels of decomposition stratified so that you can view the system at any single level and get a consitant view
- standard techniques

Priciples

- **Abstraction**: hiding implementation details, abstract, interface
- **Encapsulation**: attribute and functions are contained in a single object
- access =>
  - Information hiding, hiding internal data from other classes
- **Polymorphism**
  - Overloading
  - Overriding
- **Modularity**
  - Breaking down of a component and reassembling it
  - design measure
    - coupling
    - chesion
    - information hiding
    - seperation of concern

- SOLID Principles
  - Liskov substitution
  - Interface Segregation
  - Dependency Inversion
  - Open Close → close for modification
  - Single Responsibility
    One class do only one thing

Liskov substitution => Every sub class should be substitution for their parent class

Interface segregation => A client should never be forced to implement an interface that it doesnt use

Dependency Inversion: High level class, should not depend upon low level classes instead both ~~dependuse~~ abstraction

AI

24-10

- ML
  1. Supervised Learning
  2. Unsupervised Learning
  3. Reinforsment Learning
  4. Semi - ~~sup~~ supervised Learning