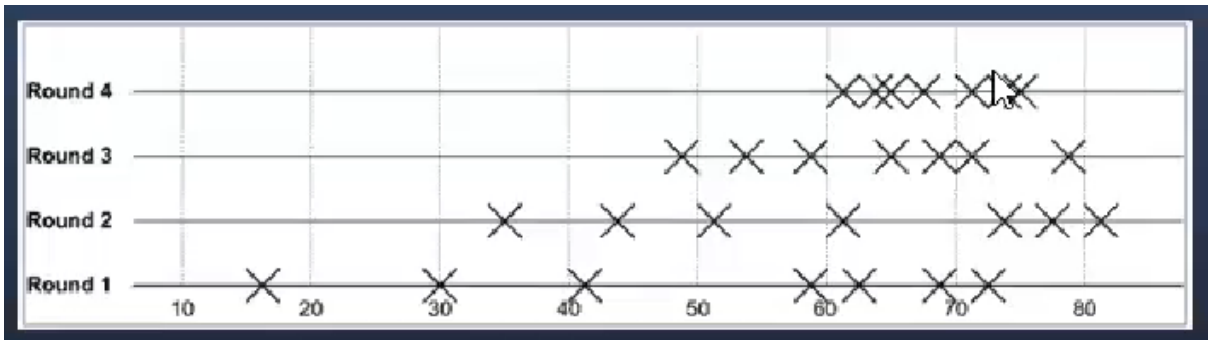
PROJECT ESTIMATION

- Project success depends on:
    - Must be on time
    - Must be within budget
    - Must be of required quality
- What is estimation?
    - Estimation covers time and budget parts and resource allocation which are for project managers not for engineers.
    - Initially, project manager sets expectation based on experience from other projects and then with time this timeline gets refined
    - Estimations must be realistic such as delivery after deadline, causing distrust from stakeholders
- Elements of sound estimate:
    - Before estimations, manager should have:
        - WBS, or activity list leading to finished product
        - Estimating effort for each task
        - Assumptions for estimate
        - Consensus that estimate is accurate
    - Assumptions make estimations accurate, because brainstorming assumptions and listing them down will help clearify the chosen estimate
- Wideband Delphi:
    - It is generic technique to measure schedule timeline and also size estimation of software
    - Background:
        - Manager chooses sub-team from team (3-7 members) whose task is to estimate and it is leaded by a moderator. These teams arrange meetings in which they have the complete WBS or task or user stories which are selected one by one. All the business and technical aspects and risks of this task is identified and estimation is then asked from members. This is repeated until all estimations converge.
    - Steps:
        - Choose Team:
            - Manager should not be modertor as he makes his own rough timeline or asks for the final timeline for

review. If he is the moderator, then this may cause biasness in decisions as the manager will defend his decisions. Hence, manager is not a moderator

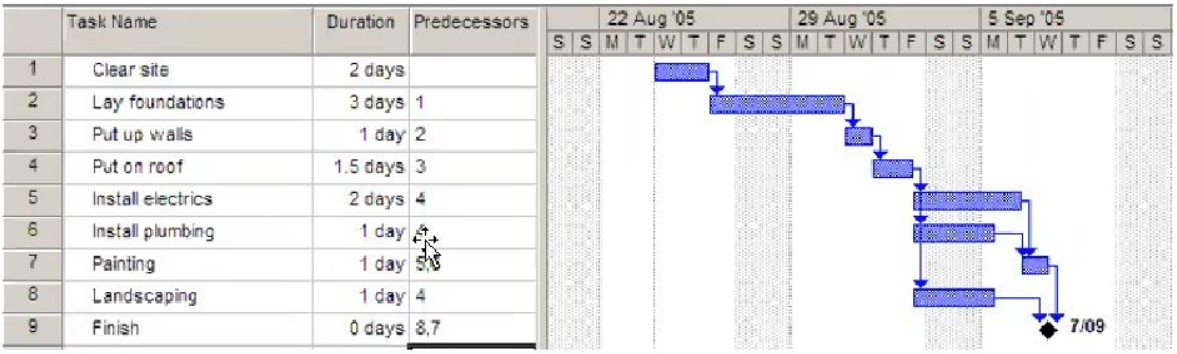
- Kickoff meeting:
  - It is preparatory meeting
  - The team formed will do meeting to:
    - Read vision and scope document
    - Brainstorm and writedown assumptions
    - Generate a WBS about 10 – 20 tasks
    - Agree on an estimate
- Individual preparation:
  - Each member will write an estimate of effort and underlying assumptions he took on a sheet provided by the moderator and then submit it
- Estimation Session:
  - Moderator now discusses all the estimations with the team members
  - All the estimates of the members are plotted, and iteration is repeated
  - The iteration (rounds) are repeated until team members disagree on revising their estimates again (cannot converge more)



- Assemble Tasks:
  - All the final tasks are then combined along with their estimates and underlying assumptions
- Review Results:
  - The assembled task list is reviewed with the team members

- GANTT and PERT:

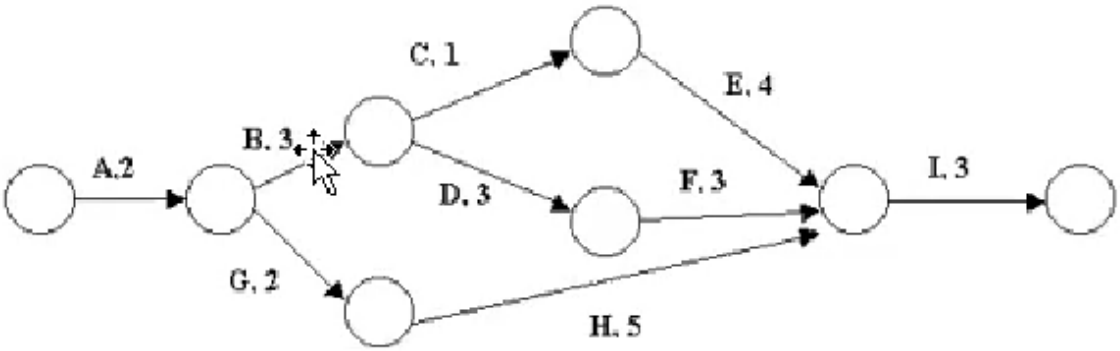
- Schedule techniques, timeline visualization
- Visual representation of tasks, their duration, dependencies, progress of project easing working of product manager
- Gantt Basics:
  - Timeline of tasks
  - Consist of bars which represent task
  - Length of bar duration of task
- Making a Gantt Chart:
  - Steps:
    - List task in project (WBS, listing down all the activities of project)
    - Add task durations (estimations after wideband delphi)
    - Add dependencies indicated by arrows (which task depend on the other to start). 0 days mean it is a milestone. Concurrent activities as 5,6,8 start right after 4 ends



- Find the critical path:
  - Sequence of tasks that take longest time to complete
  - Task on critical path are known as **critical task.** Also, they don't have **time slack**
  - This critical path is then considered the minimum time it will take to accomplish project. Hence, time slack is kept enough in order to accommodate unforseen risks.
  - **Slack / Float:** amount of time a task can be extended before it affects other tasks. OR

amount of time, changing which may not affect the deadline.
    - Hence, critical task does not have slack as affecting them, will affect deadline.

- PERT Chart:
    - Similar to Gantt chart, however, length of task bar concept rarely applies to PERT charts
    - PERT Flavours:
        - Activity on arrow:
            - Tasks and their durations are shown by arrows.
            - Arrows also show the dependencies
            - Circles represent the start and end of task (we can write dates in it)



        - Activity on node:
            - The tasks and durations are now on the nodes, instead of arrows
    - After time estimations are calculated and recorded through wideband delphi, then we get estimates as:

Activity	Predecessor	Time estimates			Expected time
		Opt. (*O*)	Normal (*M*)	Pess. (*P*)	
A	—	2	4	6	4.00
B	—	3	5	9	5.33
C	*A*	4	5	7	5.17
D	*A*	4	6	10	6.33
E	*B*, *C*	4	5	7	5.17
F	*D*	3	4	8	4.50
G	*E*	3	5	8	5.17

    - Optimistic and pessimistic are min and max times agreed upon by members. Manager then, calculates the expected time
    - Expected Time: (T_E = (O+4M+P)/6) best estimated time to complete task accounting all factors and risks that can occur
- Network Activity Diagram:
    - Terms:
        - Early start: earliest time that activity can start
        - Early finish: earliest time that activity can finish
        - Late start: latest (toleratable) time that activity can start
        - Late finish: latest time that activity can finish
        - Total float: how long activity can be delayed without delaying project completion date. (TF = LS – ES or LF - EF). It is zero for all critical path nodes
        - Free float: how long acticvity can be delayed without affecting early start of successor activity (FF = $ES_{successor}$ – $EF_{activity}$)
        - Buffer: time strategically placed on critical path prior to delivery date to accomdate for unforseen circumstances
        - Lead: acceleration of successor activity i.e the successor activity can be started before the completion of predecessor activity. Such as C requires A to finish, but if C is partially dependent, then data from A can be supplied to C before completion and C can start at 4^{th} instead of 5^{th} etc.

- Lag: delay in start of successor activity maybe due to late start of predecessor or any other factor so early start is delayed hence C instead of 10th now starts at 12th.

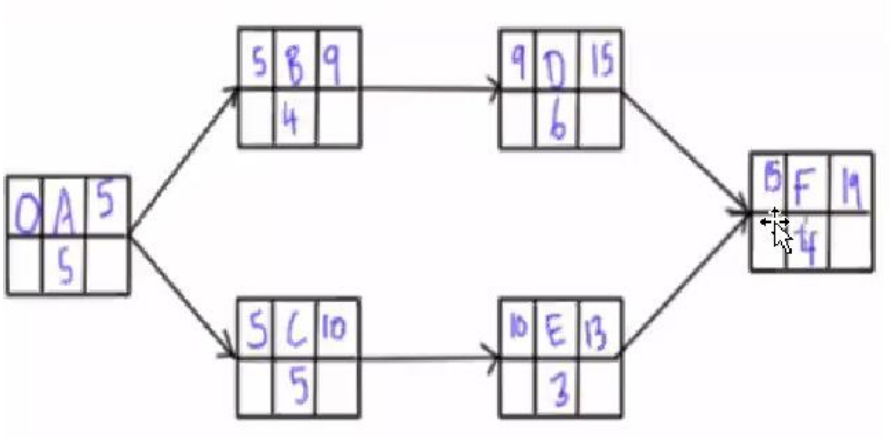
- Activity network:

Activity	Predecessor	Duration
A	-	5
B	A	4
C	A	5
D	B	6
E	C	3
F	D,E	4

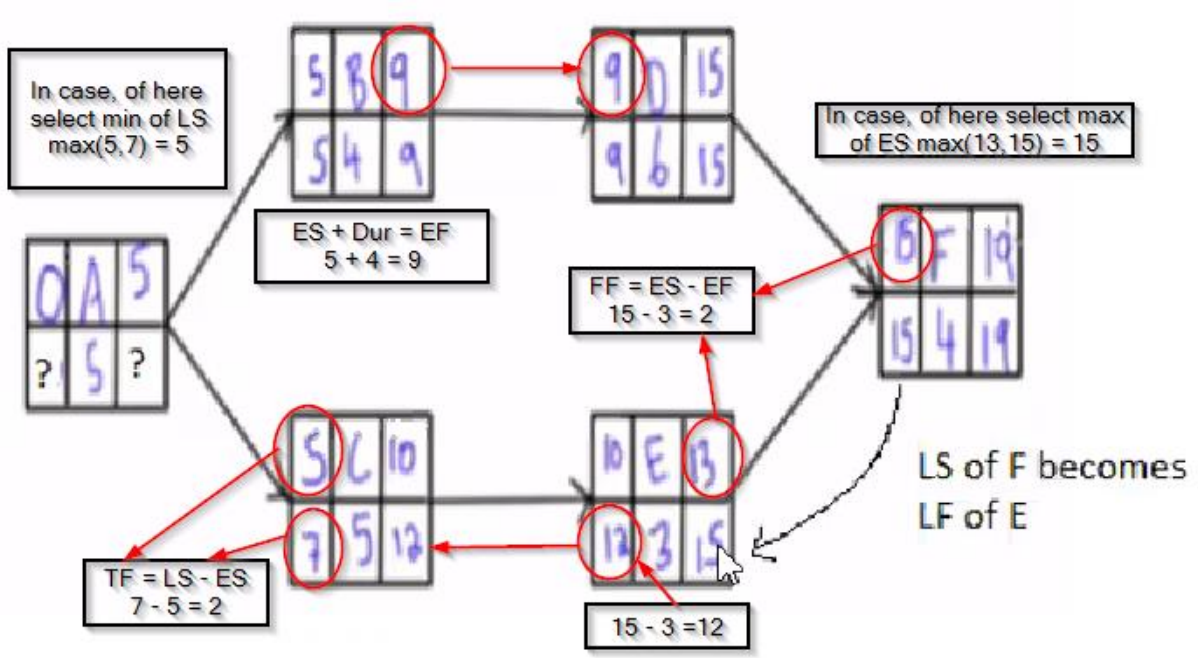
- Make a table of activities, predeccesors(dependencies) and their duration.
- Make each box like:

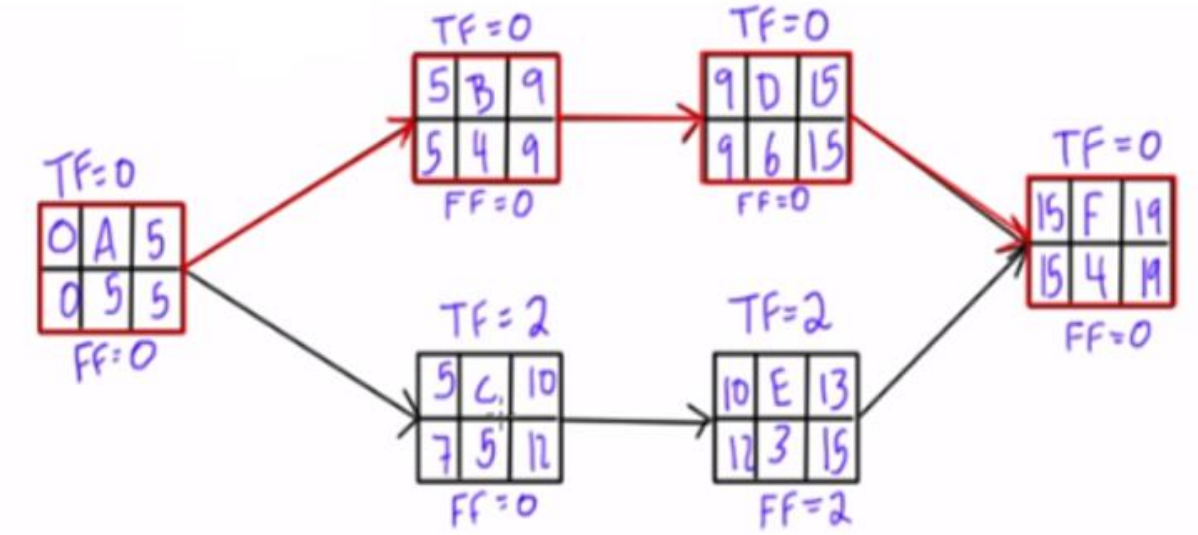
Early start	Activity name	Early Finish
Late start	Duration	Late finish

- Early finish = early start + duration



- We have two paths: ABDF, ACEF. Hence, add up all the durations. The path with longest time will be **critical path**.
- Path with TF = 0 wil be the critical path





- LOC and FP (Lines of code, functional points):
- LOC:
    - Size estimation of software as they can be used for estimating effort and cost
    - Same optimistic, pessimistic and normal estimations are estimated with the consensus of estimation team and then S = (S_{opt} + 4S_{normal} + S_{pess}) / 6 used to calculate the estimated size
    - Hence, LOC are estimated and estimated LOC for all tasks are summed up and multiplied with pay per line ($13 per line etc.) to calculate cost of developing whole software

- However, its unfair means of measuring as some lines may be simple while others might be complex and might require more time. Moreover, it also varies based on different programming languages. So LOC does not measure complextiy leading unfair calculation of efforts. Hence, we shift to FP

- Calculation of LOC:
  - Get the table of optimistic, average and pessimistic loc for all the tasks from the member consensus.
  - Calculate estimated LOC based on the formula of S
  - Sum all the LOC.
  - Cost = Cost_per_LOC * Estimated LOC

Function	Estimated LOC
User interface and control facilities (UICF)	2,300
Two-dimensional geometric analysis (2DGA)	5,300
Three-dimensional geometric analysis (3DGA)	6,800
Database management (DBM)	3,350
Computer graphics display facilities (CGDF)	4,950
Peripheral control function (PCF)	2,100
Design analysis modules (DAM)	8,400
Estimated lines of code	*33,200*

-

- FP:
  - No relation to LOC
  - Calculates efforts based on functionality provided by application of each task using 5 parameters:
    - Number of external inputs (EP):
      - Input from user or other applications e.g write, update, delete
    - Number of external output (EO):
      - Derived data created by application from the input providing information to the user e.g reports, error

messages, interface options i.e outside the application boundary

- Number of external inquiries (EI):
  - These are online input that result inimmediate response of software
  - Can be in form of both input or output. Also, involve such error message if input field empty or wrong type of data inserted or sensor data etc.
  - Examples of EIs include:
    - Data entry by users.
    - Data or file feeds by external applications.
- Number of internal logical files (ILF):
  - Files maintained by software at backend such as database files, config files etc. Maintained within application boundary
- Number of external interface files (EIF):
  - Application data from 3rd party APIs.
  - Hence not maintained within your application boundary

Information Domain Value	Count		Weighting factor Simple	Average	Complex	
External Inputs (EIs)		×	3	4	6	=
External Outputs (EOs)		×	4	5	7	=
External Inquiries (EQs)		×	3	4	6	=
Internal Logical Files (ILFs)		×	7	10	15	=
External Interface Files (EIFs)		×	5	7	10	=
Count total						

To compute function points (FP), the following relationship is used:

$$FP = \text{count total} \times [0.65 + 0.01 \times \Sigma (F_i)]$$

- This table is filled and the formula mentioned below is used to calculate FP
- Weighting factors is set independently for each organization hence, varies.

- Summation of F_i is calculated using 14 factors:

Information domain value	Opt.	Likely	Pess.	Est. count	Weight	FP count
Number of external inputs	20	24	30	24	4	97
Number of external outputs	12	15	22	16	5	78
Number of external inquiries	16	22	28	22	5	88
Number of internal logical files	4	4	5	4	10	42
Number of external interface files	2	2	3	2	7	15
Count total						*320*

Factor	Value
Backup and recovery	4
Data communications	2
Distributed processing	0
Performance critical	4
Existing operating environment	3
Online data entry	4
Input transaction over multiple screens	5
Master files updated online	3
Information domain values complex	5
Internal processing complex	5
Code designed for reuse	4
Conversion/installation in design	3
Multiple installations	5
Application designed for change	5
Value adjustment factor	**1.17**

1. Does the system require reliable backup and recovery?
2. Are specialized data communications required to transfer information to or from the application?
3. Are there distributed processing functions?
4. Is performance critical?
5. Will the system run in an existing, heavily utilized operational environment?
6. Does the system require online data entry?
7. Does the online data entry require the input transaction to be built over multiple screens or operations?
8. Are the ILFs updated online?
9. Are the inputs, outputs, files, or inquiries complex?
10. Is the internal processing complex?
11. Is the code designed to be reusable?
12. Are conversion and installation included in the design?
13. Is the system designed for multiple installations in different organizations?
14. Is the application designed to facilitate change and ease of use by the user?

- Calculation FP:

- Multiply the weighted factor of each EIs, EOs, EQs, ILFs, EIFs identified from the project with their respective count
- Est. count is calculated in the same way as formula S in LOC.
- FP Count = Est. count * weight
- Count total = $\sum$ (FP Count)
- Value adjustment factor = 0.65 + (0.01 * $\sum$ (F_i)) where F_i is the weighted value (1 to 5) for all the 14 factors
- In above case VAF = (0.65 + 0.01 * 52) = 1.17
- FP = count total * VAF
- Cost_per_FP = FP_per_month * labor_rate_per_month (may be per month or per year but must be consistent for both variables. This has to be calculated if not given)
- Cost = FP * cost_per_FP