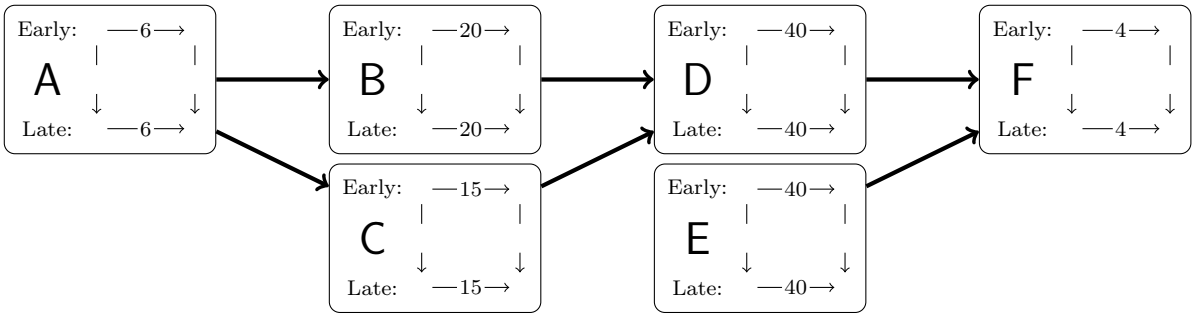
Your problem

Task	Duration	Finish first
A:	6 min	Nothing
B:	20 min	A
C:	15 min	A
D:	40 min	B, C
E:	40 min	Nothing
F:	4 min	D, E

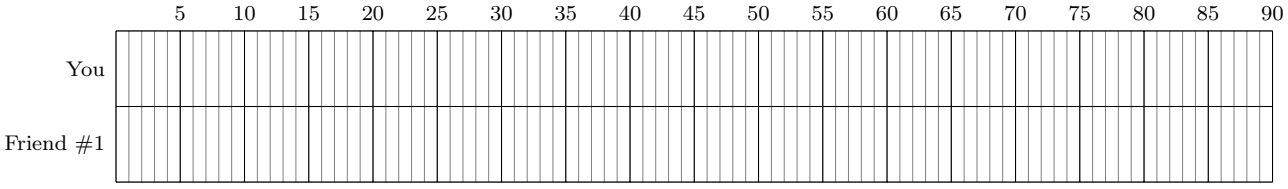
Fill in the times in the corners of each task bubble. Start $\xrightarrow{\text{Duration}}$ Finish



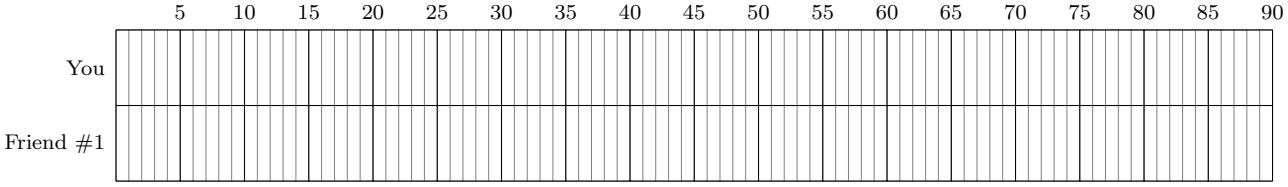
Order the tasks into a priority list based on their float time, from highest priority to lowest.

Priority list: _____

You only have two workers. Place the tasks on the time-line **according to the priority list**.



Now place the tasks where you think they should go:



Critical path analysis

Critical path analysis optimisitically assumes we have enough resources to complete tasks as efficiently as possible. It calculates how quickly each task can be completed, and then the "float" of each task: how much each task can be delayed before it delays the entire project. In the real world, resources are tight, and so the float is used to prioritize which task should be worked on with the limited resources available. Tasks with no float, should be done ASAP, while tasks with lots of float can be put off without delaying the project as a whole.

Instructions

(Early) The "early" rows are the earliest time the task can start and finish. This is calculated from left to right. The earliest start time is the maximum of the earliest finish times of the tasks that need to be finished first. If no task needs to be finished first, then it can be started immediately, time = 0. The earliest finish time is the earliest start time plus the duration.

(Late) The "late" rows are the latest time the task can start and finish without delaying the optimistic project completion time (which is 70 minutes in this case). This calculated from right to left (backwards). The latest finish time is the minimum of the the latest start times of the tasks that depend on this one being finished. If no task depends on it being finished first, then it can be finished at the very end (70 minutes in this case). The latest start time is the latest finish time minus the duration.

(Float) The vertical arrows have the "float". This is calculated as the difference between the earliest start time and the latest start time (or the earliest finish time and the latest finish time; you get the same number both times).

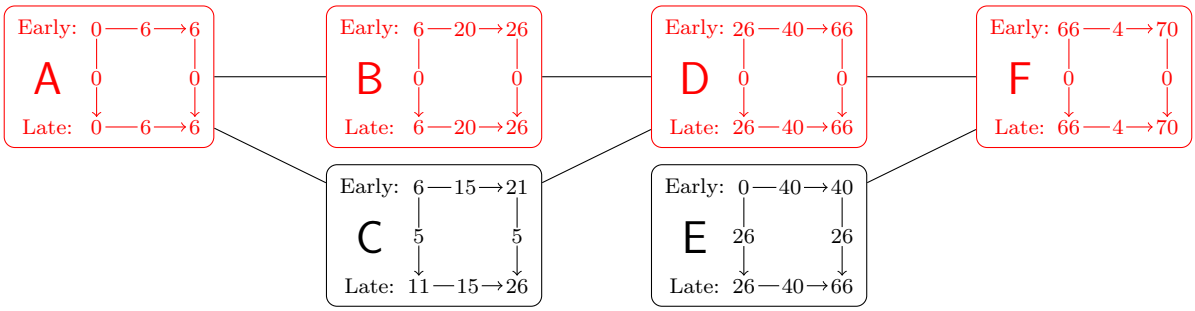
(Priority) The vertical arrows have the float. Tasks with float = 0 are called **critical** and must be done as soon as possible to avoid delaying the project. Tasks with small float should be higher priority than tasks with large float. Put the tasks in order from highest priority (lowest float) to lowest priority (highest float). If two tasks have the same priority, I usually put an equals sign between them, and if one task has a higher priority I put a $>$ greater than sign $>$ between them.

(First timeline) In this timeline, place tasks onto the timeline following these four rules: (1) you cannot overlap tasks, (2) a task cannot start before all of its "finish first" tasks are finished, (3) if a task can be started, then you must start a task, and (4) in case more than one task can be started, choose the one with the highest priority (from the priority list) first. That means that once the priority list, durations, and "finish first"s are given, there is only one correct timeline.

(Second timeline) However, that priority list might give a cruddy timeline. So in this timeline, you only have to obey the first two rules. So whenever a task finishes, you can choose any task to start (obeying rule 2, but possibly disobeying 4) or even just chill out and don't start anything (disobeying 3).

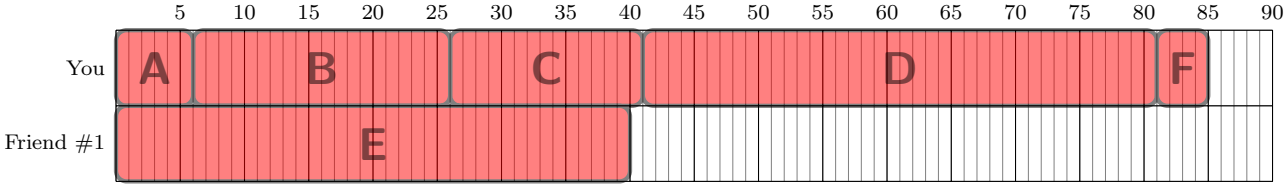
Answers

Here are the exactly correct answers to the first three. If your answer is different, then it is probably wrong.



Priority list: $A = B = D = F > C > E$.

You can scramble A, B, D, F if you want. So $F = D = B = A > C > E$ is also right.



Using our basic critical path analysis we get that the project could have been completed in 70 minutes with unlimited resources, but if we use the float time to prioritize tasks we end up taking 85 minutes.

Better answer

The problem is that we were too impatient to start tasks. We started a very long duration and very low priority task (E), and that prevented us from doing shorter, higher priority tasks a few minutes later. Here is a better (best possible) schedule: