**COMP10050 Assignment 2**

**Group BO:**

Martina D’Argenzio 22789139

Jack Dunne 22483576

https://csgitlab.ucd.ie/jack-dunne626/bo-assignment-2.git

inserting text here

**dev build**:

rather than getting started right off with diagram, only list it line by line in plaintext

e.g.: don’t get started on the diagram until after the tasks and such are working

otherwise you’d spend too much time debugging the diagram rather than, yknow, doing the actual fuckin project

“recursive dependencies”, if a task has a dependency and that dependency has a dependency itself, then recursively move through it.

base case no dependencies

recursive case dependencies, evaluate (think about this more plz)

**assignment briefing:**

objective: make a plan and a set timeline with responsibilities to be discussed later

(should be planning more than programming.)

Ideally 4 separate arrangements where work is done, with achievable set objectives that **must** be completed by the end of each arrangement

With breaks too. No burnouts.

**Notes:**

Max length of anything is to be 100chars (try not go past 100 spaces in the program)

Get the length of one line and then determine how long a string can be, to save the formatting.

**Objective 1: (max 10 tasks)**

Ask the user to either view an example chart with 10 tasks, or create their own series of tasks

**Viewing example chart:** to create the table

(sorry for this.) format the table using the example provided

**Task display:**

Declare variable max-length for string. Update variable with string length of inputted task name if length > variable. When printing, printf("%\*s |", max\_length, taskname);.

ask to edit, test, or quit (objective 2) (strcmp edit test quit != 0, invalid)

**Create own series of tasks:** using a struct with name, start/end month, and dependencies

How many tasks? (0 invalid, else iterate for number of tasks entered)

Task name (malloc string)

Start/end month (input <= 0 or > 12, invalid input)

Number of dependencies (0 = skip, else iterate for number of dependencies entered)

Dependent task(s) (input <= 0 or > task number, invalid input)

# Optional tasks

Add \a to every invalid input. Scare people!