**COMP10050 Group BO Assignment 2**

**Group BO:**

Martina D’Argenzio 22789139

Jack Dunne 22483576

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

**How is the Gantt displayed?**

# How to map the task to struct?

**How did you edit and change tasks?**

**How did you implement search for a circular dependency?**

Throughout design, we continuously planned and reoriented ourselves by setting goals to be accomplished each time we worked in collaboration. This was done to make the project as simple, efficient, and workable as possible.

The Gantt chart was initially going to be displayed via file handling (a file named template with the format of the chart) however this raised a few problems. Firstly, the template would be “constant” in that there would be 10 rows displayed, no matter the amount of tasks given. Secondly, the method of which the Gantt chart would be displayed was to be character-by-character which, while being simple, is both inefficient and borderline unreadable.  
Instead we opted to print the Gantt chart in-house, which resulted in a much cleaner chart because only the number of tasks given would be displayed through rows.

We kept a dev branch for experimentation and crash-testing new ideas. Successful builds were pushed to master.

Our ASCII art is a portrait of Karl Marx. Jack is reading the Communist Manifesto and thinks it’s a good read, so he’s went and spread his socialist beliefs.

**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

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

base case no dependencies, return 0

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

global int var “dependencyCheck”, if > 3 exit (return to thing).

**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:**

Invalid entry function should immediately exit the program.

**Definitions:** kept in definitions.h

MAX\_LENGTH = 32

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

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

Function: **Viewing example chart:** to create the table, clear the screen first

**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);.

**Placeholder task names:**

Lorem / ipsum / dolor / sit / amet / consectetur / adipiscing / elit / sed / do

Global char array “months” of size 12, (malloc of size 4 for each entry???),

(“Jan”,”Feb”,”Mar”,”Apr”,”May”,”Jun”,”Jul”,”Aug”,”Sep”,”Oct”,”Nov”,”Dec”);

Definition: global char EMPTY = “” (for table row 0 col 0)

Size of months columns are to be 5 spaces wide, 1 space padding on either side of the month at row 0, and 2 spaces on each side of the “X” to mark a task.

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

Function: **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!