Link to Gitlab repository:
https://csgitlab.ucd.ie/LabibaJaigirdar/bs-assignment-2/-/tree/main, Group BS, Assignment 2

How is the Gantt displayed?

- The gantt is displayed by printing the input to a file.
- This file is then read and output to the console using the function printGantt().

How is the task mapped to the struct?

- The struct has the parameters:
    - Char taskName (contains the name of the task)
    - Int startMonth (the number of the month during which the task begins)
    - Int endMonth (the number of the month during which the task reaches completion)
    - Int numOfDependencies (the number of dependencies that the task has)
    - Int dependencies[] (an array containing the IDs of each dependency)
- Two arrays of structs (task[] and testGantt[]) were created to hold the information of up to 10 tasks entered by the user, and the information contained in the test gantt, respectively.

How did you edit tasks?

- I edited tasks by getting the user to input the name of the task they wish to edit, and comparing the task name using strcmp() to each task in the struct to find the correct task in the array of structs. If the name didn't match, the user is prompted to re-enter the name.
- Then the user is prompted to enter a new task name, start month, end month, number of dependencies, and dependencies for the selected task.
- This overwrites the information of the previous task.
- The edited gantt is then printed.

How did you implement a test for a circular dependency?

- I implemented a test for a circular dependency by creating a recursive function called printDependentTasks.
- This function takes in a struct task, an array of which tasks have been visited, and the index of the task to be tested.
- The function marks the current task as visited by assigning it a 1 in the array visitedTasks[].
- Then, using a for loop, the code loops through the dependencies of the current tasks
- dependentTaskID is assigned the index of the dependency.

- If the task has already been visited (i.e. it is assigned a 1 in the array), “Circular dependency found!” is printed and the variable circularFound is set to 1.
- Otherwise, printDependentTasks is called recursively to check the dependencies of the current dependency, and so on, until either a circular dependency is found, or each dependency has been checked.
- If no circular dependency is found in the current path, the last task in the path is set to 0 (i.e. reset to not having been visited).
- The function returns circularFound. If circularFound is 0, “No circular dependency found.” is printed.

What’s your ASCII art about?

- My ascii art is official art of my favourite character from a rhythm game I play called Project Sekai. (this is the art: https://static.miraheze.org/projectsekaiwiki/6/64/Kanade_24_trained_art.png)