GCIS-123
Class Activity & Problem Solving
#Bonus Activity
List and dictionary

Goals of the Assignment

You have just started work at ***Gostco*** as a senior developer for the ***Cineplexo*** project. One of your tasks is to set up an application that will enable you to schedule the tasks of the developers of your team, in order to complete and minimize the development process time.

As a senior developer, for each T_i task, you know exactly which T_j tasks need to be completed beforehand. Therefore, you know that you need to schedule task T_i after tasks T_j.

Consider the following scenarios:

1. **Scenario 1:** Consider the following project that consists of three tasks T_1, T_2, and T_3. The dependence among the tasks is as follows:
   - T_3 : T_1, T_2
   - T_2 : T_1
   - T_1 :

   You know that to complete task T_3, tasks T_1 and T_2 must first be completed. And to complete task T_2, task T_1 must be completed. T_1 is independent of any of the tasks. To complete all tasks your application will schedule your tasks in order as follows: [T_1, T_2, T_3]. This result means you need to complete first T_1, then T_2 and finally T_3. This schedule is valid since all the tasks will be scheduled (T_1, T_2, and T_3).
2. **Scenario 2:** Consider the following project that consists of three tasks T_1, T_2, and T_3. The dependence among the tasks is as follows:
   - T_3 : T_2
   - T_2 : T_1, T_3
   - T_1 :

   Your application will be unable to schedule any tasks. It will return an empty schedule [] since you have circular dependence between T_3 and T_2. To complete T_3, you need to complete T_2 and to complete T_2 you need to complete T_3 and T_1 which is illogically.

3. **Scenario 3:** Consider the following project that consists of four tasks T_0, T_1, T_2, and T_3. The dependence among the tasks is as follows:
   - T_3 : T_2
   - T_2 : T_1

- T_1 :
- T_0 :

Your application can schedule your tasks either [T_0, T_1, T_2, T_3] or [T_1, T_0, T_2, T_3]. Both schedules are valid since they include all the tasks of the project.

4. **Scenario 3:** Consider the following project that consists of four tasks T_1, T_2, T_3 and T_4. The dependence among the tasks is as follows:
   - T_4 : T_3
   - T_3 : T_2, T_1
   - T_2 : T_1
   - T_1 :

   Your application will schedule your tasks as follows [T_1, T_2, T_3, T_4]. This schedule is valid since it includes all the tasks of the project.

Before Starting

Create a new python module in a file named: **yourGroupNumber-activity4.py** and add to this module the logic in the following section (change yourGroupNumber by your correct group).

Activities

a. [5%]. Define the dependence between tasks as dictionary of lists, `tasks_dependencies`, where each key represents a task and the value of this key is a list of its prerequisite tasks. Your application will use later this dictionary to schedule your task.

   For example, for the following tasks:

   - T_3 : T_2, T_1
   - T_2 : T_1
   - T_1 :

The constructed `tasks_dependencies` dictionary will include three entries. For example, the entry T_2: [T_1] means to complete T_2 (key) you need to complete first the list of tasks [T_1]. If a key (i.e task) has no dependency (e.g, T_1 in the example above), its value will be an empty list.

b. **[20%].** Define a function `mirror` that takes a dictionary as parameter (e.g. `tasks_dependencies`), swap the keys and values and returns the results in a new dictionary (e.g., `dependency_tasks`). More precisely, in the generated

dictionary each key represents a task and the value of this key is a list of tasks that depend on this key.

For example, from the *tasks_dependencies* of the following tasks:

- T_3 : T_2, T_1
- T_2 : T_1
- T_1 :

The *mirror* function will generate the dictionary, *dependency_tasks*. It is clear that this dictionary includes two keys: T_2 and T_1. The value for the key T_1 is [T_2, T_3] which means that T_2 and T_3 depend on T_1.

c. **[20%].** Define a function *remove_value* that takes a list of tasks, *list_tasks*, a specific task T_x, and a dictionary of lists (e.g. *tasks_dependencies*) as parameters. The tasks in *list_tasks* are keys in the dictionary of lists (e.g. *tasks_dependencies*). For every task in *list_tasks*, this function will remove T_x from its values in *tasks_dependencies*. This function will also return a new list including the keys in *tasks_dependencies* that have empty values **after the removing step**.

For example, assume you have generated already the *tasks_dependencies* from the following tasks:

- T_3 : T_2, T_1
- T_2 : T_1
- T_1 :

Thus, by calling *remove_value(['T2', 'T3'] , 'T1' , tasks_dependencies)* the task T_1 will be removed from the values of the keys T_2 and T_3 in the *tasks_dependencies*. Only, the value of the task T_2 becomes empty (an empty list), then the *remove_value* will return a list including this task T_2 only.

d. **[35%].** Using the functions implemented in b. and c., define a function *schedule* that takes a dictionary of lists (e.g. *tasks_dependencies*) as parameter and returns an ordered list of tasks so that all tasks will be done. If no scheduling can be done between tasks, *schedule* should return an empty list of tasks (or *None*).

Pseudo-code for the schedule algorithm:

1. **Put all tasks with no pre-requisites into a startBy list.**
2. **Add one task from starBy to your schedule**
3. **For each task C in the startBy list, find each task D which have C as a prerequisite and remove C from its list.**
4. **Add D to the startBy list and go to 1.**

Submission Instructions

1. Rename correctly yourGroupNumber-activity3.py. Otherwise -10 % will be applied.

2. Use the best practice in python to define your variables and style your code. Otherwise -10 % will be applied.

3. Include the appropriate internal-documentation (i.e. comments & docstring). Otherwise -10% will be applied. In the docString of your module add the names of all the members of your team.

4. Upload **ALL the file** to the MyCourses Assignment box as (Activity04.zip). Otherwise -10 % will be applied (only one team-member needs to submit to MyCourses).

5. Be sure that you have pushed **ALL the file** to your GitHub classroom repository (**for each team)**. Otherwise -10% will be applied. DON'T WAIT UNTIL THE LAST DAY TO CHECK YOUR GITHUB AND FIX YOUR PROBLEMS.

6. **Reminder:** Note regarding group work/group projects: all students in a group are considered a single unit and will be held accountable for any offense perpetrated by the group as a whole or an individual member of that group. Note that there should be no distinction in cheating incidents between the student that provided the material and the student that received the material. Both must receive the same consequences.