Middle East Technical University **Department of Computer Engineering**

CENG 315

Algorithms

Fall 2024-2025

Take-Home Exam 5

Due date: 8 December 2024, Sunday, 23.59

1 Problem Definition

You are working on an international project as an undergraduate researcher. The project is currently in the proposal phase and you are trying to plan the work items. Each work item can either be by itself or is connected to one or more items, and may need to be completed either before or after the item that its connected to.

However, it is not easy to order the work items as is since some items depend on each other either directly or indirectly. So, to combat this planning issue, you decide to combine such dependent work items into a work package where the item-based planning will be handled later on.

Given the dependencies between work items as a graph, you are expected to identify the work items that need to be combined into a work package, replace the dependent work item nodes with one work package node, and find an ordering of the things that need to be done in the project.

```
std::vector< std::vector<int> > find_work_order(
        const std::vector< std::vector<bool> > &dependencies);
```

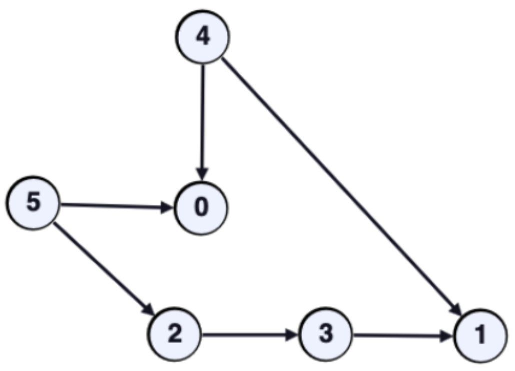
In this exam, you are expected to implement the function `find_work_order`. The dependencies between the work items are given to you in `dependencies` as a matrix. You should identify the work item groups where each item is directly or indirectly dependent on each other, and abstract them away as a work package. After this abstraction, you should find an ordering of the work items and work packages and return the `work_order` accordingly.
It should be noted that:

- If the completion order does not matter for two work items, you should assign higher priority to the work item with the smaller ID.
- When comparing the work packages, you should consider the work item with the smallest ID in that work package to be the ID of that work package and decide the ordering accordingly.
- You should be able to capture indirect dependencies where a work item is indirectly depending on itself. However, a work item may also directly depend on itself.
- Each work item can at most belong to one work package (i.e. each work item can at most be part of one direct/indirect dependent group).
- Ordering of the work items in a work package is not important.

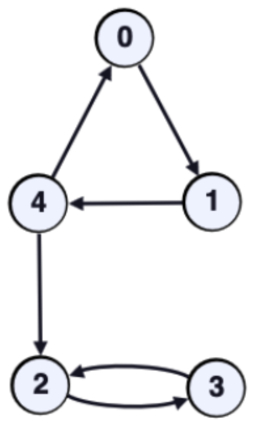
1.1 Example I/O

```
INPUT
─────
dependencies:
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 1 0 0
0 1 0 0 0 0
1 1 0 0 0 0
1 0 1 0 0 0

OUTPUT
──────
work_order:
[4] -> [5] -> [0] -> [2] -> [3] -> [1]
```

```
INPUT
─────
dependencies:
0 1 0 0 0
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
1 0 1 0 0

OUTPUT
──────
work_order:
[0, 1, 4] -> [2, 3]
```



(a) dependency graph of first I/O



(b) dependency graph of second I/O

2 Limits and Specifications

In this exam, the complexity of your implementations will be checked by the run of your implementations staying within the VPL limits. Hence the system and variable limitations are as follows:

- at least 4, at most 100 `work_items` in the `dependencies` graph,
- 16 seconds as the maximum execution time,
- a maximum memory limit of 1 GB,
- and a stack size of 4 MB for function calls (ie. recursive solutions)

Additionally:

- You will implement your solutions in the `the5.cpp` file. Do not change the first line of `the5.cpp`, which is `#include "the5.h"`.
- Do not change the arguments and the return value of the given functions in the file `the5.cpp`, but you are free to add other functions to `the5.cpp`. However, do not include any other library or write include anywhere in your `the5.cpp` file (not even in comments).
- You are given a `test.cpp` file to test your work on ODTUCLASS or your locale. You can and you are encouraged to modify this file to add different test cases.
- You can test your `the5.cpp` on the virtual lab environment. If you click run, your function will be compiled and executed with `test.cpp`. If you click evaluate, you will get feedback for your current work and your work will be graded with a limited set of inputs.
- If you want to test your work and see your outputs on your locale you can use the following commands:

```
> g++ test.cpp the5.cpp -Wall -std=c++11 -o test
> ./test
```

3 Regulations

- **Implementation and Submission:** The template files are available in the Virtual Programming Lab (VPL) activity called "THE5" on ODTUCLASS. At this point, you have two options:
  - You can download the template files, complete the implementation, and test it with the given sample I/O on your local machine. Then submit the same file through this activity.
  - You can directly use the editor of the VPL environment by using the auto-evaluation feature of this activity interactively. Saving the code is equivalent to submitting a file.
- **Programming Language:** You must code your program in C++. Your submission will be tested on the VPL environment in ODTUCLASS, hence you are expected to make sure your code runs successfully there.
- **Cheating: This assignment is designed to be worked on individually.** Additionally, the use of any LLMs (chatgpt, copilot, the other one that you are thinking about...) and copying code directly from the internet for implementations is strictly forbidden. Your work will be evaluated for cheating, and disciplinary action may be taken if necessary.
- **Evaluation:** Your program will be evaluated automatically using "black-box" testing, so make sure to obey the specifications. No erroneous input will be used. Therefore, you don't have to worry about invalid cases.