Graph

Instructions for students:

- Complete the following classes/methods on Heap.
- You may use any language to complete the tasks (Java / Python).
- All your methods **for an individual task** must be written in one single **.java** or **`.py`** or **`.ipynb`** file. DO NOT CREATE separate files for each task.

NOTE:

- **YOU CANNOT USE ANY BUILT-IN FUNCTION EXCEPT** len **IN PYTHON. [negative indexing, append is prohibited]**
- **YOU HAVE TO MENTION SIZE OF ARRAY WHILE INITIALIZATION**
- **YOUR CODE SHOULD WORK FOR ALL RELEVANT SAMPLE INPUTS**

Dear Students, you have been given instructions and driver code for the last 6 labs. For the last two labs of this semester, no driver code will be given. You will develop everything (necessary functions, class, driver code, etc.) in your preferred language (Java or Python).

To solve the problems, draw a graph according to your preference. However, your graph must have a minimum of **7 vertices** and a minimum of **16 edges**. **`Include a picture of your graph with your submission.`**

`Note:` `Solve all the problems for both adjacency matrix representation and adjacency list representation. You can write different functions for each task to make your life easier.`

Task 1:

Given an undirected, unweighted graph as input, write a function to find the vertex with maximum degree and return the degree of that vertex.

Task 2:

Given an undirected, edge-weighted graph as input, write a function to find the vertex whose sum of edge weights is maximum.

Task 3:

Solve Task 1 and Task 2 for directed, edge-weighted graphs considering only outgoing edges.

Task 4:

W rite a function that will receive a directed graph weighted and convert it to an undirected weighted graph. Consider the weight of the edges will be unchanged.

Note: As there are 4 tasks and you must complete them for both adjacency list and adjacency matrix, you have to complete 8 tasks in total.