Instructions: Please note that handwritten assignments **will not be graded**. Use the provided LaTeX template to complete your homework. Please do not alter the order or spacing of questions (keep each question on its own page). When you submit to Gradescope, you must mark which page(s) correspond to each question. **You may not receive credit for unmarked questions**.
When including graphical figures, we encourage the use of tools such as graphviz or packages like tikz for simple and complex figures. However, these may be handwritten only if they are neat and legible (as defined by the grader).

List any collaborators (besides TAs or professors) here: N/A

1. (30 points) [W6, ★★★★★] Graph Modelling. [**Note: Similar to the maze project, this is a graph modeling question.**] Consider the following problem:

   You are given a connected weighted graph G that represents a road network connecting n cities. An edge (i, j) in G means that there is a road segment from city i to city j. The weight of edge (i, j) denoted by l_{ij} is the length of that road segment, which is *assumed to be an integer*. In addition, let p_i denote the price of one unit of fuel at each city i (assume that there are no gas stations on road segments and that fuel can only be purchased in cities). Finally, let C (also an integer) denote the fuel tank capacity of your car. Determine the cheapest trip cost from start city s to end city e. Your car uses one unit of fuel per unit of distance traveled and starts with an empty tank. You may choose to fill up your fuel tank by any integer amount as long as you don't exceed C.

   Describe a graph model (vertices and edges) that will enable one call of **unmodified** Dijkstra's to solve this problem. **Note:** we recommend that you define some notation to help you thoroughly describe your model.

   (a) (5 points) Describe what each vertex of your graph represents. (Hint: there will be at most $n \times (C + 1)$ nodes in your model.)

   Each vertex represents a configuration of the car's fuel level $(0 - C)$ and the city the car is in. That is, for every city n, there will be $C + 1$ vertices representing the fuel level at that city. The vertex will be represented as (k, f) where k is the city and f is the fuel level.

   (b) (15 points) Describe the conditions necessary for an edge to exist between two vertices in your graph. (Hint: think about actions that might change the state of your model.)

   An edge exists between two vertices (i, f) and (j, g) if the the following conditions are met:

   - There is a road segment between the two cities i and j in graph G.
   - The fuel level f is greater than or equal to the distance between the two cities l_{ij}.

- The fuel level $g = f - l_{ij}$.
- The fuel level f is less than or equal to C and the fuel level g is greater than or equal to 0. *This should be taken care of by the node generation process.*

There will also be edges from (i, f) and (i, g) represented by the action of refueling at a city if the following conditions are met:

- Where the fuel level g represents all integers greater than f.
- The fuel level f and g are less than or equal to C. *This should be taken care of by the node generation process.*

(c) (10 points) Describe how you would calculate the cost of an edge in your graph. Please provide a general mathematical expression (formula) for the edge weights. Note you might have different formulas for different edges, make sure to describe the conditions under which you would use each formula.

The cost of an edge in the graph is the cost of the fuel used to travel from one city to another. The cost of the edge (i, f) and (j, g) is given by the formula:

$$\text{cost} = 0$$

this is because the cost of the fuel is occurs in the city i not during transit.

The cost of the edge (i, f) and (i, g) is given by the formula:

$$\text{cost} = (g - f) \times p_i$$

where p_i is the price of fuel at city i and $g - f$ is the amount of fuel used to refuel at city i.

2. (15 points) [W6, ⋆⋆] Provide an example of a directed graph with a negative weight edge, and a specific start and end node such that a Dijkstra's search (that is, Dijkstra's algorithm with the modification that it terminates after the end node is deleted from the priority queue) will **not** give the correct answer. Your example shouldn't need more than a handful of vertices.

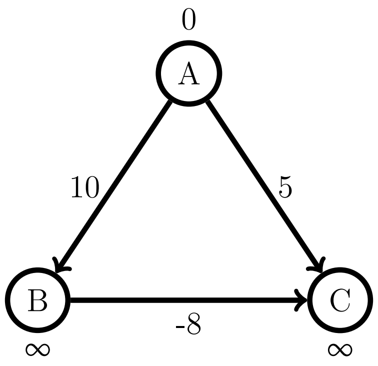

Figure 1: Example of a directed graph with a negative weight edge.

In the graph in Figure 1, if we start at node A and end at node C, Dijkstra's algorithm will not give the correct answer. The algorithm will choose the path $A \to C$ with a total weight of 5, but the correct shortest path is $A \to B \to C$ with a total weight of 2.

This is because after the first iteration of Dijkstra's algorithm, the algorithm will traverse to nodes B and C and set their distances to accordingly, as shown in Figure 2. Dijkstra's will then check the priority queue and find that the next node to visit is C. The algorithm will then terminate and return the shortest path to C as the shortest path.

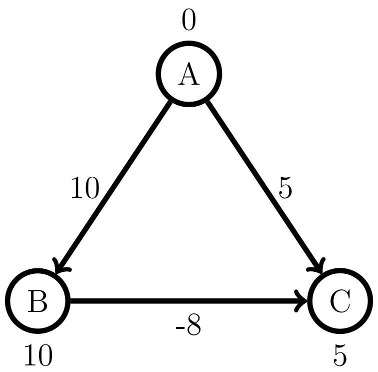

Figure 2: First iteration of Dijkstra's algorithm.

3. (15 points) [W6, ⋆⋆⋆] Assume Bellman-Ford has identified the presence of a negative-weight cycle (in the second for loop in the algorithm in the slides). Explain how to identify and output the vertices that constitute this cycle.

   This can be done simply by traverseing through the predecessor array.

```
def find_negative_cycle(pred, start):
    cycle = []
    current = start
    while True:
        cycle.append(current)
        current = pred[current]
        if current == start:
            break
    return cycle
```

   Where `pred` is the predecessor array and `start` is the vertex that is the start of the cycle. This function will return a list of vertices that constitute the negative-weight cycle.

4. (10 points) [W7, ⋆] For the following questions, select the true statement(s). **No explanation is necessary for these questions.**

   (a) What is true about the residual graph in the Ford-Fulkerson algorithm?

   ■ The sum of residual capacities over all edges in the residual graph is equal to the sum of the capacities in the original graph.

   □ The number of edges in the residual graph is at least equal to the number of directed edges in the flow network.

   ■ The capacity of any edge in residual graph does not exceed the capacity of the corresponding edge in the flow network, if it exists.

   □ The capacity of an edge in the residual graph depends on the direction, quantity, and capacities of the pipes between the two vertices.

   (b) What must be true if there is a directed path in the residual graph from the source to the sink in the Ford-Fulkerson algorithm?

   □ The flow is not maximized.

   ■ The path is an augmenting path.

   □ There is a cut in the original flow graph whose capacity is equal to the flow in the network.

5. (10 points) [W7, ⋆] Flow. For the following questions, select whether the statement is true or false, and write a *brief* explanation of your reasoning.

   (a) The maximum flow in a flow network is equal to the minimum cut of the network.
   □ True [✓ ■ False]

   This is false because the maximum flow in a flow network is the maximum amount of flow that can be pushed through the network. The minimum cut of the network is the minimum capacity of the edges that need to be removed to disconnect the source from the sink. These two values are not necessarily equal.

   Max flow is equal to the minimum cut if and only if the flow is maximized and the residual graph has no augmenting paths from the source to the sink.

   (b) In the residual graph of the maximum flow, there are no augmenting paths from the source to the sink.
   [✓ ■ True] □ False

   This is true because if there are no augmenting paths from the source to the sink, then the flow is maximized. (ie there is no more flow that can be pushed through the network.)

   (c) In a residual network, the capacity of an edge can be zero.
   [✓ ■ True] □ False

   This is true because when the capacity of an edge is zero in a residual network, it means that the edge is saturated and no flow can pass through it. (The edge is effectively removed from the residual.)

   (d) If a single edge's capacity in a flow network is increased, the maximum flow value can only either stay the same or increase.
   [✓ ■ True] □ False

   This is true because the maximum flow value is the maximum amount of flow that can be pushed through the network. You can't decrease the max flow by making an edge more capable of carrying flow.

6. (20 points) [W7, ⋆⋆⋆] How can standard, unmodified max-flow algorithms (which only work with a single source and single sink node) be applied to a graph with multiple sources and sinks? Describe the necessary steps or considerations.

   To run unmodified max-flow algorithms, like Ford-Fulkerson, on a graph with multiple sources and sinks, we can create a new master source and sink node and connect the original sources and sinks to the new master source and sink. The capacity of these edges will infinity or at least greater than or equal to the total capacity of the original source/sink it connects. This will create a single source and single sink graph that can be used with the unmodified max-flow algorithm.