Q1 Modified Max Flow
25 Points

Consider a flow network $G = (V, E)$ where every edge has capacity 1. Let f be a st-max-flow vector (where f_e denotes the flow through edge $e \in E$).

Now let's suppose that we remove from edge e_r from the graph. It might be possible that the current flow f is no longer a valid maximum flow. Given G, s, t, f, e_r, design an $O(|V| + |E|)$ time algorithm that returns a new st-maximum flow f' that is a valid on graph G', where G' is the modified graph of G with edge e_r removed. **Provide a runtime analysis and proof of correctness in your answer.**

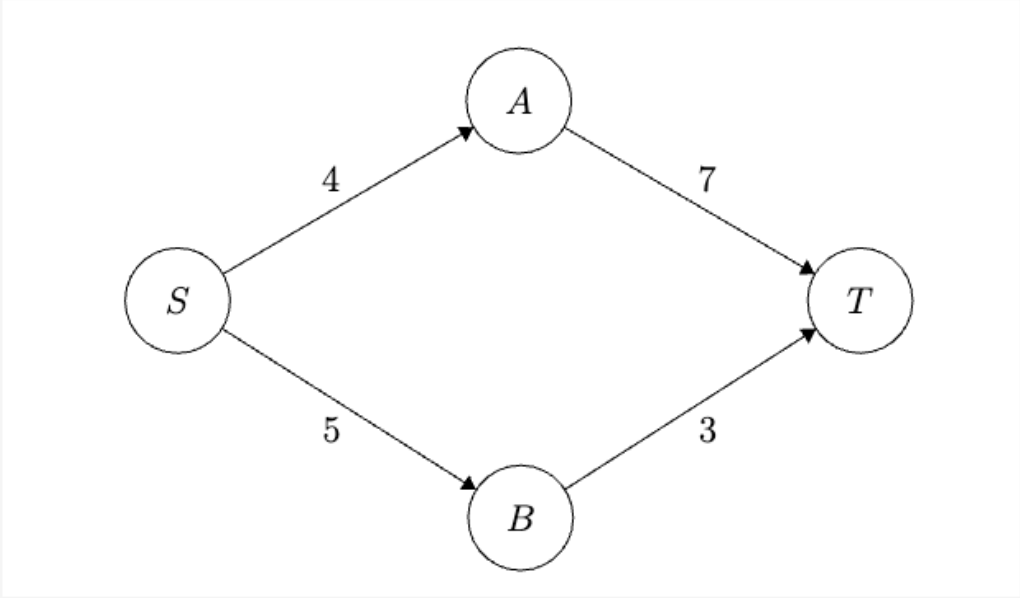
Step 1: Let Gf denote the residual graph of G with respect to the initial flow f. Initialize f' = f
Step 2: Remove the edge e, from Gf, adjusting the residual capacities accordingly.
Step 3: Use a standard augmenting path algorithm (e.g., BFS or DFS) to find paths from s to t in the modified residual graph G. Adjust the flow f' along the augmenting paths until no augmenting path exists.
Step 4: Return f', the new maximum flow.

Runtime:
Constructing the initial residual graph Gf takes O(|V| + |E|).
Removing e, and updating residual capacities takes O(1).
Finding augmenting paths in G and updating the flow involves O(|V| * |E|) operations in the worst case.
Hence, the total runtime is O(|V| * |E|), dominated by the cost of finding augmenting paths.

Correctness:
Gf, the residual graph, accurately represents the capacities available for flow adjustments at each step. Removing e, ensures no flow can pass through it.
Adjusting the flow f' using augmenting paths ensures that flow conservation is maintained and that the final flow saturates all available capacity from s to t.
By the Ford-Fulkerson theorem, the algorithm terminates with a valid maximum flow.

Q2 Max Flow
25 Points

Consider the following graph $G = (V, E)$:



Recall the formulation of max flow in a network as a linear program from lecture. You will convert this flow network to an LP.

Constraints:
Capacity:
$0 \leq f_{S,A} \leq 4$,
$0 \leq f_{S,B} \leq 5$,
$0 \leq f_{A,T} \leq 7$,
$0 \leq f_{B,T} \leq 3$.
Flow Conservation:
$f_{S,A} = f_{A,T}$
$f_{S,B} = f_{B,T}$

Objective Function:
Max $(f_{S,A} + f_{S,B})$

Q2.2 Find Matrices
5 Points

Find the $m \times n$ matrix A, $m \times 1$ vector b, $n \times 1$ vector c (where m is the number of constraints and n is the number of variables) that represents the above LP in standard form. You can type matrices using the LaTeX \bmatrix environment. Use this link to see how to type it into the textbox.

A =

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

B=

$$\begin{bmatrix} 4 \\ 5 \\ 7 \\ 3 \\ 0 \\ 0 \end{bmatrix}$$

C=

$$\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Q2.3 Solve the Linear Program
6 Points

Using any method, solve the linear program. Show your work.

f S,A = f A,T
f S,B = f B,T

f S,A = min (f S,A , f A,T)
f S,A = min (4 , 7)
f S,B = min (f S,B , f B,T)
f S,B = min (5 , 3)
f S,A = 4
f S,B = 3
Max (f S,A + f S,B) = 7

Q2.4 Dual
7 Points

Compute the Dual and solve it. Show your work. What does the dual represent?

1. Let y1, y2, y3, y4 be the dual variables for the capacity constraints:
fS,A4, fS,B≤ 5, ƒA,T ≤7, fBT ≤ 3.
2. Let λ1 and λ2 be the dual variables for the equality (flow conservation) constraints:
fS,A - fA,T = 0, fS,B - fB,T = 0

Objective:
Minimize w = 4y1+5y2 + 7y3 + 3y4
Constraints (derived from primal variables):
1. y1 - λ1 ≥ 1
2. y2 - λ1 ≥ 1
3. -λ1 + y3 ≥ 0
4. -λ2 + y4 ≥ 0
To minimize w = 4y1+5y2 + 7y3 + 3y4, set y1=1,y2=1,y3=0,y4=0.
w = 4(1)+5(1) + 7(0) + 3(0)
w=9

The dual represents the minimum cut in the flow network. The value w=9 corresponds to the total capacity of the edges in the minimum cut. By the max-flow min-cut theorem, the value of the maximum flow (primal solution) equals the value of the minimum cut (dual solution). Hence, the maximum flow is z=7.

Q3
0 Points

This part of the problem set will done on a separate Gradescope assignment, but here are general instructions.

You will submit all your code on a separate **Problem Set 9 (Coding)** Gradescope assignment where it will then be verified by an autograder. Python, Java, C++ are accepted. You can resubmit as often as you want (until the deadline). Each question will have a corresponding code file that you will implement your algorithm in, and you will upload all code files on Gradescope for grading.

Gradescope will automatically test your submissions. You can resubmit as often as you want (until the deadline) and you can see on Gradescope how many test cases you currently passed. Your submission must pass all test cases on Gradescope, but the final pass/fail decision is made by a TA. (This is to make sure your submission is actually solving the problems using the Monte Carlo method. For example, printing hardcoded values or using a deterministic algorithm is obviously not a valid solution.). To reiterate, **you must use the Monte Carlo method to solve these questions**. Any other methods will receive no credit.

File Structure, Input, and Output
For each question, we give a description on the file name specifications and how to read the input and write the output for each problem. The programs should read input from standard input and prints its output to standard output. In Python, this can be done using `sys.stdin` and `print`. In Java, this is `System.in` / `System.out.println`. In C++, this is `cin` and `cout`.

Allowed Libraries
Here are the allowed libraries/module for each language:
Python: `math`, `random`
C++: `bits/stdc++.h`
Java: `java.util.*`

Q3.1 Monte Carlo: Integral Estimation (25 points)
0 Points

Problem Description
Using the Monte Carlo method, write a program that takes in as input a lower bound value a and an upper bound value b (where a could possibly be `-INF` and b could be `INF`) and outputs the following integral:

$$\int_a^b e^{-x^2} dx$$

Hint: To handle infinity values, you can look into this.

File Name: `Integral.py` (python), `Integral.cpp` (C++), `Integral.java` (Java)

Input Specification: The lower bound value a in one line and the upper bound value b in another line. You can assume that $a \leq b$. Examples:

```
-4.0
7.0
```

```
-INF
7.1
```

```
-INF
INF
```

Output Specification: The program must print the estimated numerical value of the integral. It should not print anything else.

Grading: There will be 20 randomly generated test cases for this problem. A test case is considered passing if the absolute difference between your code's output (`actual`) and the expected output (`expected`) is within the following tolerance:

```
abs(actual-expected) <= max(0.03 * max(abs(actual), abs(expected)), 1e-4)
```

The amount of points you earn for this question will be proportional to the percentage of test cases you passed. (Note: it is okay if you occasionally fail one or two test cases. You can keep resubmitting until you pass all test cases.)

Q3.2 Monte Carlo: Perimeter of Ellipse Estimation (25 points)
0 Points

Problem Description
Using the Monte Carlo method, write a program that takes in as input six coefficients A, B, C, D, E, F and outputs the perimeter of the ellipse defined by the following equation:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0.$$

The ellipse will always be within a 300 x 300 rectangle centered at (0, 0).

Hint: Researching on algorithms to find a bounding polygon on a set of points may be helpful here.

File Name: `Ellipse.py` (python), `Ellipse.cpp` (C++), `Ellipse.java` (Java)

Input Specification: The six coefficients of the ellipse equation each in separate lines (where A is the first line, B is the second line, etc.). **You may assume that the equation will always define an bounded ellipse**. Example:

```
4.0
2.1
5.3
2.2
1.1
-1.0
```

Output Specification: The program must print the estimated numerical value of the perimeter of the ellipse defined by the provided coefficients. It should not print anything else.

Grading: There will be 10 randomly generated test cases for this problem. A test case is considered passing if the absolute difference between your code's output (`actual`) and the expected output (`expected`) is within the following tolerance:

```
abs(actual-expected) <= max(0.03 * max(abs(actual), abs(expected)), 1e-4)
```

The amount of points you earn for this question will be proportional to the percentage of test cases you passed. (Note: it is okay if you occasionally fail one or two test cases. You can keep resubmitting until you pass all test cases.)