Question 1 – Integer Programming

14 marks total

A breakfast cereal company is planning the delivery of wheat and corn to its three factories over the next six months. Each month they use a single ship for deliveries with a capacity of 4000 tonnes. They have estimated the demands (tonnes) at each factory and the costs ($/tonne) of purchases each month, as shown in the Python file. Wheat and corn on hand at the end of a month can be stored for a cost of $1.50 per tonne. Suppose they currently have 600 tonnes of wheat and 300 tonnes of corn in storage at each factory and would like to have these amounts remaining at the end of the six months.

a) Formulate a linear programming problem to determine the optimal schedule for delivery of wheat and corn to the three factories. Implement your model in Python, using comments to briefly explain each variable and constraint. *[6 marks]*

b) Currently the company pays $12,000 to use the ship in each month so they might be able to reduce costs by not using it every month. Furthermore, there is a cost of $5,000 to visit one of the factories in a month (so the total cost to visit all three factories in one month would be $27,000). Extend your model to an integer programming problem that includes these ship costs in determining the optimal schedule for delivery of wheat and corn to the three factories. Implement your model in Python, using comments to briefly explain each variable and constraint. *[5 marks]*

c) Suppose the ship has 5 holds, each with a capacity of 800 tonnes. In practice, wheat and corn need to be transported in separate holds so that they do not mix. Extend your integer programming problem to include this requirement. Implement your model in Python, using comments to briefly explain each variable and constraint. *[3 marks]*

When run, your Python code should produce the optimal objective value for each one of these three models. For parts (b) and (c), your code should additionally show when the ship is being used.

Question 2 – Metaheuristics

6 marks

Suppose you have a collection of 100 square pieces, all of which have a number on each edge. You want to lay the pieces out on a 10-by-10 board so as to minimise the sum of the absolute differences between touching edges. The pieces cannot be rotated.

The Python file gives you an initial layout of the pieces on the board and the **BoardCost** and **RunSA** functions. Use these to demonstrate how Simulated Annealing can be used to solve this layout problem. You should define the neighbourhood of a solution (as a comment in Python) and then write appropriate **ChooseNeighbour** and **Move** functions. You will then need to experiment to find suitable parameters for the number of iterations and the cooling factor.

When the code you submit is executed it should run the Simulated Annealing procedure with the final parameters you have selected and display the total cost of the solution found. Your code should not take more than 30 seconds to run.

END OF EXAMINATION