5.17 A2 MODULE 4737: DECISION MATHEMATICS 2 (D2)

Preamble

Knowledge of the specification content of Modules *C1* to *C4* and *D1* is assumed, and candidates may be required to demonstrate such knowledge in answering questions in Unit *D2*.

The specification content of this module is to be understood in the context of modelling real-life situations, and examination questions may ask for comment and interpretation, including where appropriate cross-checking between a model and reality.

Game Theory

Candidates should be able to:

(a) understand the idea of a zero-sum game and its representation by means of a pay-off matrix;

(b) identify play-safe strategies and stable solutions;

(c) reduce a matrix by using a dominance argument;

(d) determine an optimal mixed strategy for a game with no stable solution

(i) by using a graphical method for $2\times n$ or $n\times 2$ games, where $n=1, 2$ or 3,

(ii) by converting higher order games to linear programming problems that could then be solved using the Simplex method.

Flows in a Network

Candidates should be able to:

(a) represent flow problems by means of a network of directed arcs, and interpret network diagrams;

(b) find the optimum flow rate in a network, subject to given constraints (problems may include both upper and lower capacities);

(c) understand the meaning of the value of a cut, use the maximum flow-minimum cut theorem and explain why it works;

(d) introduce a supersource or supersink in networks with more than one source or sink, and replace a vertex of restricted capacity by a pair of unrestricted vertices connected by a suitable flow;

(e) use a labelling procedure, with arrows showing how much less could flow in each direction, to augment a flow and hence determine the maximum flow in a network.

Matching and Allocation Problems

Candidates should be able to:

(a) represent a matching problem by means of a bipartite graph;

(b) use an algorithm to find a maximal matching by the construction of an alternating path;

(c) interpret allocation problems as minimum-cost matching problems;

(d) use the Hungarian algorithm to find a solution to an allocation problem, including the use of a dummy row or column, use the covering method to check whether a matching is maximal, and augment and interpret a revised cost matrix.

Critical Path Analysis

Candidates should be able to:

(e) construct and interpret activity networks, using activity on arc;

(f) carry out forward and reverse passes to determine earliest and latest start times and finish times, or early and late event times;

(g) identify critical activities and find a critical path;

(h) construct and interpret cascade charts and resource histograms, and carry out resource levelling.

Dynamic Programming

Candidates should be able to:

(i) understand the concept of dynamic programming, working backwards with sub-optimisation;

(j) use stage and state variables, actions and costs;

(k) set up a dynamic programming tabulation and use it to solve a problem involving finding a minimum, maximum, minimax or maximin.