| | | | |
|---|---|---|---|
| **Module code** | SM-1203 | | |
| **Module Title** | Discrete Mathematics | | |
| **Degree/Diploma** | Bachelor of Science (Mathematics) | | |
| **Type of Module** | Major Core | | |
| **Modular Credits** | 4 | **Total student Workload** | 10 hours / week |
| | | **Contact hours** | 4 hours / week |
| **Pre-requisite** | A-level Mathematics or equivalent | | |
| **Anti-requisite** | None | | |

**Aims**

To introduce concepts and techniques of mathematics (mostly discrete) needed for various areas of Mathematics and Computer Science. On completing this module, the student should be able to use the concepts and techniques of discrete mathematics to prove programme correctness, investigate algorithm complexity, set up economical computer networks and model computation.

**Learning Outcomes**

*On successful completion of this module, a student will be expected to be able to:*

| | | |
|---|---|---|
| Lower order: | 30% | - describe the logic statements and write formal proof<br>- describe sets, multi-sets and set of natural numbers |
| Middle order: | 60% | - understand counting techniques for counting permutations, combinations, bit strings and subsets |
| Higher order: | 10% | - describe minimum-weight spanning trees and minimum-distance trees |

**Module Contents**

- Logic statements, connectives, canonical forms (CNF, DNF), inference, formal proof, predicates.
- Sets, multi-sets; set of natural numbers, basic properties; countable sets.
- Counting techniques for counting permutations, combinations, bit strings and subsets.
- Relations: representations and manipulation through Boolean matrices, digraphs; Posets, Hasse diagrams.
- Functions and their growth.
- Algorithms: proof of correctness; computational complexity.
- Graphs: vertex, edge; incidence, adjacency matrices, vertex degree; complete bipartite graphs; subgraphs, graph isomorphism; paths, cycles, connected graphs; Euler circuits, Hamilton cycles.
- Trees: minimum-weight spanning trees, minimum-distance trees.
- Modelling computation: languages, grammars; finite-state machines; Turing machines.

| | | |
|---|---|---|
| Assessment | Formative assessment | Weekly feedback, tutorial and discussion |
| | Summative assessment | Examination: 60% |
| | | Coursework: 40%<br>- 2 Class tests (40%) |