COURSE CSE1010: COMPUTER SCIENCE 1

Level: Introductory

Prerequisite: None

Description: Students explore hardware, software and processes. This includes an introduction to the algorithm as a problem-solving tool, to programming languages in general and to the role of programming as a tool for implementing algorithms.

Parameters: Access to an appropriate computer work station, the Internet, a programming language/environment and associated support materials. It is recommended that the course be taught in tandem with one or more programming courses.

Supporting Courses: CSE1110: Structured Programming 1
CSE1120: Structured Programming 2, and/or any
Intermediate project course involving imperative programming

Outcomes: The student will:

1. identify and describe the nature, approaches and areas of interest of computer science

1.1 define and describe computer science with consideration of:

1.1.1 the main goal of the discipline

1.1.2 the use of algorithms

1.1.3 computer systems used to test and/or implement algorithms

1.1.4 the translation of algorithms through programming

1.2 describe the general areas of interest of computer science including:

1.2.1 the theory of computation

1.2.2 algorithms and data structures

1.2.3 programming methodology and languages

1.2.4 computer elements and architecture

1.2.5 human–machine and machine–machine interfacing

1.2.6 automata

1.2.7 artificial intelligence

1.2.8 visual and auditory rendering

1.2.9 general development of information technology applications

1.3 compare and contrast computer science, computer engineering and information technology; e.g., theoretical versus applied, general versus specific, exploratory versus applicatory

1.4 describe some of the misconceptions associated with computer science; e.g., synonymous with programming, reliant on solitary individuals for the bulk of its advances, relatively little real-world contact, the learning of various computer applications

1.5 computer science's role in an information society

2. demonstrate an understanding of the nature, design and use of basic algorithms associated with problems involving the sequential inputting, processing and outputting of data

2.1 define algorithms and explain how they are used

2.2 compare and contrast the "iterative and incremental" and "waterfall" models of software development

2.3 demonstrate the analysis and design stages of a Systems Development Life Cycle model using appropriate tools; e.g., flowcharts, pseudocode, input/processing/output (IPO) charting

2.4 demonstrate a number of core algorithms including:

2.4.1 accumulation (keeping a running total)

2.4.2 determining the mean

2.4.3 determining minimums and maximums

3. explain and demonstrate the nature of structured programming

3.1 consider the rationale for structured programming

3.2 consider GOTO-less programming

3.3 consider three fundamental control structures—sequential, decision and iterative

4. explain and demonstrate an understanding of the nature, evolution, types and role of programming languages

4.1 describe how various programming languages have dealt with data representation; e.g., binary and hexadecimal systems, standard data types, data storage

4.2 describe the nature of programming language, specifically that these languages:

4.2.1 reflect a simplified version of natural language

4.2.2 evolved in tandem with algorithms and hardware over a number of generations

4.2.3 reflect the IPO data processing paradigm

4.3 describe and demonstrate how programming languages are used in the coding stage of a Systems Development Life Cycle model by converting a representative set of algorithms into executable code

5. explain the nature, evolution and basic architecture of a von Neumann computer system

5.1 create a block diagram of a stereotypical von Neumann machine

5.2 describe a number of typical devices associated with each block

5.3 show the flow of data through the computer under the direction of a program

6. demonstrate basic competencies

6.1 demonstrate fundamental skills to:

6.1.1 communicate

6.1.2 manage information

6.1.3 use numbers

6.1.4 think and solve problems

6.2 demonstrate personal management skills to:

6.2.1 demonstrate positive attitudes and behaviours

6.2.2 be responsible

6.2.3 be adaptable

6.2.4 learn continuously

6.2.5 work safely

6.3 demonstrate teamwork skills to:

6.3.1 work with others

6.3.2 participate in projects and tasks

7. make personal connections to the cluster content and processes to inform possible pathway choices

7.1 complete/update a personal inventory; e.g., interests, values, beliefs, resources, prior learning and experiences

7.2 create a connection between a personal inventory and occupational choices