CTI Short Learning Programme in Comprehensive Programming

Module Descriptions 2015
Computer Literacy

Module description
Knowing how to use a computer has become a necessity for many people. In order for the student to know how to use a computer, students need to understand some fundamental concepts for example; what contributes to making the computer run and operate correctly. This module introduces the computer and how it functions to students. Students will gain a thorough understanding of information and communication. Students will understand the fundamental principles of the internet and email concepts.

Word processing involves creating and editing text documents. The student will be introduced to the concepts of word processing and will understand and use a spreadsheet to organise data and present information.

Students will understand the concepts of databases and students will discover many ways in which they can, for example, sort data and represent it in a report. Students will be introduced to the concepts of presentations and understand and learn how to create a new presentation, and how to add images, charts and objects to a presentation. Students will also learn how to apply animation and transition effects to make a presentation come to life.

Processing and Logic Concepts

Module description
This module lays the foundation for an understanding of the logic concepts on which computer programming is based. The module concentrates on six key areas in particular, namely number systems, logic concepts, decision tables, system flowcharts, program flowcharts and pseudocode.

The first section will introduce students to different number systems; they will apply their knowledge to convert from one number system to another and perform basic addition and subtraction using numbers from different number systems. They will gain an overview of how the computer performs arithmetic operations, and when to use the one’s and two’s complements.

Logic concepts will give students an understanding of logic symbols and ways that these logic symbols can be incorporated into electronic circuits in the form of logic gates, which can be further represented in the form of truth tables. The module also includes Boolean equations and the ability to draw logic diagrams from these equations. De Morgan’s theorem and Karnaugh maps are used for proving and simplifying Boolean expressions. Using these logic concepts students will learn how to construct decision tables, and be introduced to the concept and functions of system flowcharts.

The final section of the module focuses on the ability to write and understand the logic of pseudocode as a means of outlining and converting a programming problem into an easily readable solution that can be applied to any high-level programming language.

Program Design

Module description
The aim of this module is to provide students with an understanding of the concepts of a computer program and how to write and analyse the structure of simple sequential programs in pseudocode. Program control statements are identified, explained and implemented in constructing simple programs with the help of one-, two-, and three-dimensional arrays. Simple sort algorithms using arrays are also taught and depicted in pseudocode. The concepts of file handling, or the opening and closing of files and how they are coded in pseudocode specifically, are also explained. The student is given practice in writing pseudocode with the help of exercises before undertaking similar questions in a test environment.
Software Engineering

Module description
The aim of this module is firstly to introduce students to business process modeling, the role of the systems analyst, and the different stages in the systems development life-cycle. This includes a discussion of the different charts, diagrams (Gantt charts, context diagrams), and tools described in the Unified Modeling Language (UML) that can be used in the analysis and design of a system.

UML makes use of the following tools to design the system: classes, objects, and class diagrams; sequence and communication diagrams, which show the interactions between objects; state diagrams, which show the operations or behaviours of a single object; activity diagrams, which model the activities of all objects in the system for a specific purpose; component diagram, which models various software-designs. The students are then given a practical introduction to UML for use as a tool in the systems development process. More specifically, students will familiarise themselves with use cases and scenarios, identify different actors that play a role in a system, and learn to draw and use case diagrams. The module also explores the use of state, sequence, collaboration, activity and deployment diagrams.

At the end of the module students will have to incorporate all taught concepts into the given projects. This project will also require user documentation. The user documentation will consist of an explanation for the use case diagram, activity diagram, class diagram, sequence diagram, communication diagram, and state machine diagram. Students will also have to incorporate their knowledge into a written theory-based exam.

Database Design Concepts

Module description
Database systems are at the core of any business organisation. The demand for more complex data structures is ever increasing as applications become ever more sophisticated. The use of databases is therefore fundamental to the development of any significant information system.

The aim of this module is to provide students with an essential basic knowledge of database systems which will enable them to design and implement effective relational database solutions. These skills will allow students to become efficient database developers or administrators using good practices and techniques. This module focuses on systems analysis, entity relationship diagrams, data normalisation and mapping a database’s design to tables.

Database Management

Module description
The aim of this module is to introduce students to the concepts of databases as well as a practical approach to developing databases. The course starts off with the fundamentals of database design by explaining the RDM (relational database model). From there on, the concepts of creating a database are covered as well as populating and implementing functions of a database.

The concepts of creating a database are: creating a database, altering a database, and creating tables. These tables are created with certain constraints such as primary keys and foreign keys. The module then looks at how to insert data into the tables that were created. Detailed explanations and exercises are given to demonstrate how to query information (i.e. retrieve information) from a database. Certain data is selected from a table as well as inserting selected data in another table. The module also explains how to update and delete information in a table. There are a few extras including indices, joins, views, stored procedures, triggers, and database security, which are all used to either retrieve or update data in a database. These are all explained to give the database and functions an edge.

At the end of the course, students will have to incorporate the concepts taught into a project. This project will also require user documentation. The user documentation will consist of author details, installation details, design details, and any extra information which may be relevant to the project.

Students will also have to incorporate their knowledge into a theoretical and practical exam. The theory will be written at the end of the course and the practical will be done after the theory exam. The practical exam tests the student’s ability to create a database and apply functions to the database.

Main Programming Language

Module description
C++ (CSLAC and CSLBC)  
or Java (JVTAC and JVTBC)  
or C# (CSPAC and CSPBC)  
or VB.NET (VBSAC and VBBC)  
or PHP (PHPAC and PHPBC)