Overview page

Introduction to imperative programming
Level: Beginners
Contact time: One evening a week for 10 weeks (there will be a 2 week break part way)

Computer programs are used by most of us in our daily lives. Software is present in most modern devices in one form or another.

This module teaches the basics of writing a procedural software program. You will gain an understanding of the constructs which are used in most software and how to write software programs.

PDF document information

Contact hours: Up to 2 hours per contact week for tutorials. Learners will also be required to watch lecture recording and carry out programming activities on their own PC.

Synopsis: Computer programs are used by most of us in our daily lives. Software is present in most modern devices in one form or another. This module teaches the basics of writing a procedural software program. You will gain an understanding of the types of programming languages available and the constructs which are used in most software. By the end of the course you will be able to design and develop your own procedural software using software libraries and your own code. You will learn how to design your own basic algorithms and write software to meet user requirements. This module will use the C programming language.

Notes: This module is aimed at professional learners who are looking to learn the basics of software development.

Assessment: Learners will be assessed on their understanding through short programming exercises submitted as a portfolio at the end of the course, and a test.

Aim: This module aims to introduce learners to procedural programming.

Learning outcomes: Design algorithms to solve a range of problems; Use control structures and I/O facilities to implement algorithms; Design and build applications using a range of programming techniques; Use the basic structure and features of a programming language; Write programs that adhere to style and documentation guidelines.

Syllabus: Types of programming language; Interpretation and compilation; Programming environments and libraries; Use of variables; Data types; Operators; Evaluation of expressions; Algorithms; Control structures; Pseudo-code; Testing; Problem solving; Variable scope; Recursion; File storage; Code layout; Documentation.