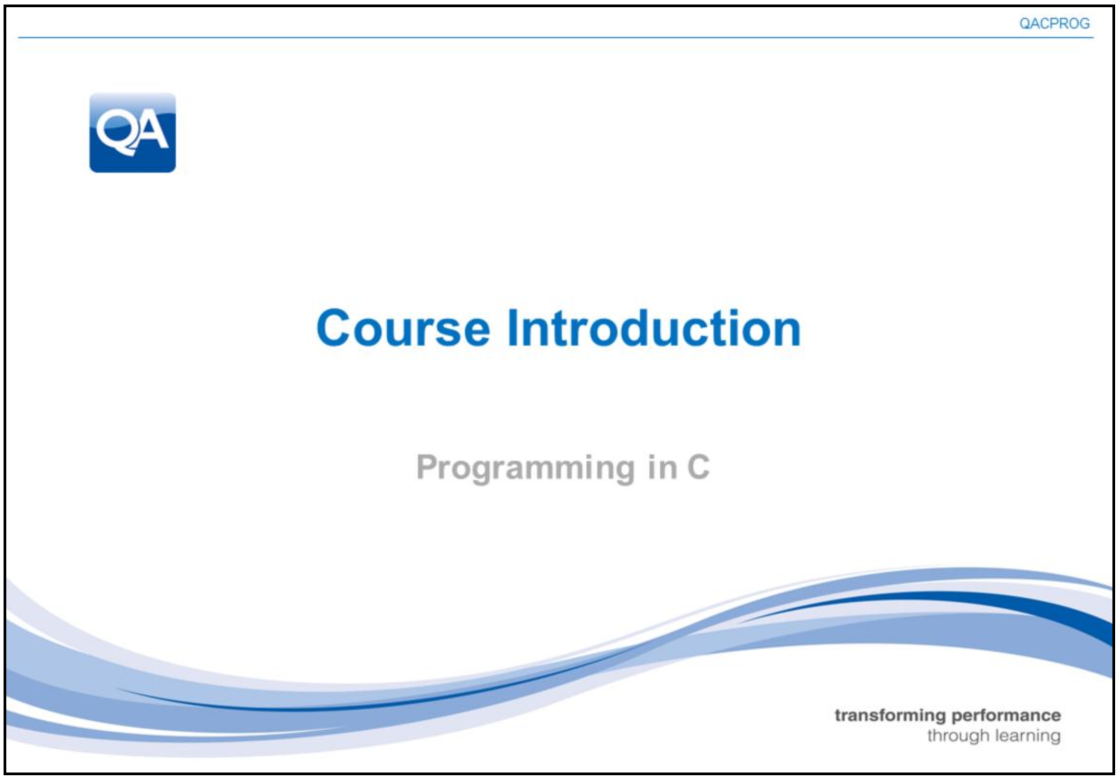
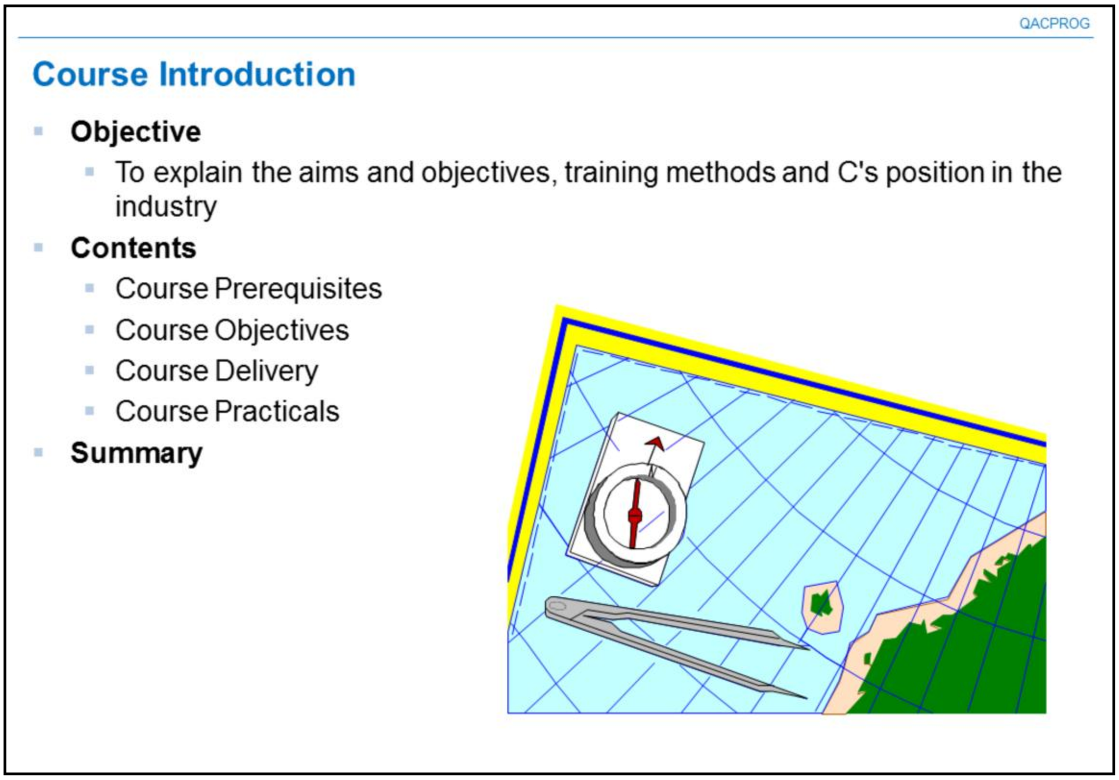


QACPROG

Course Introduction

Programming in C

transforming performance
through learning

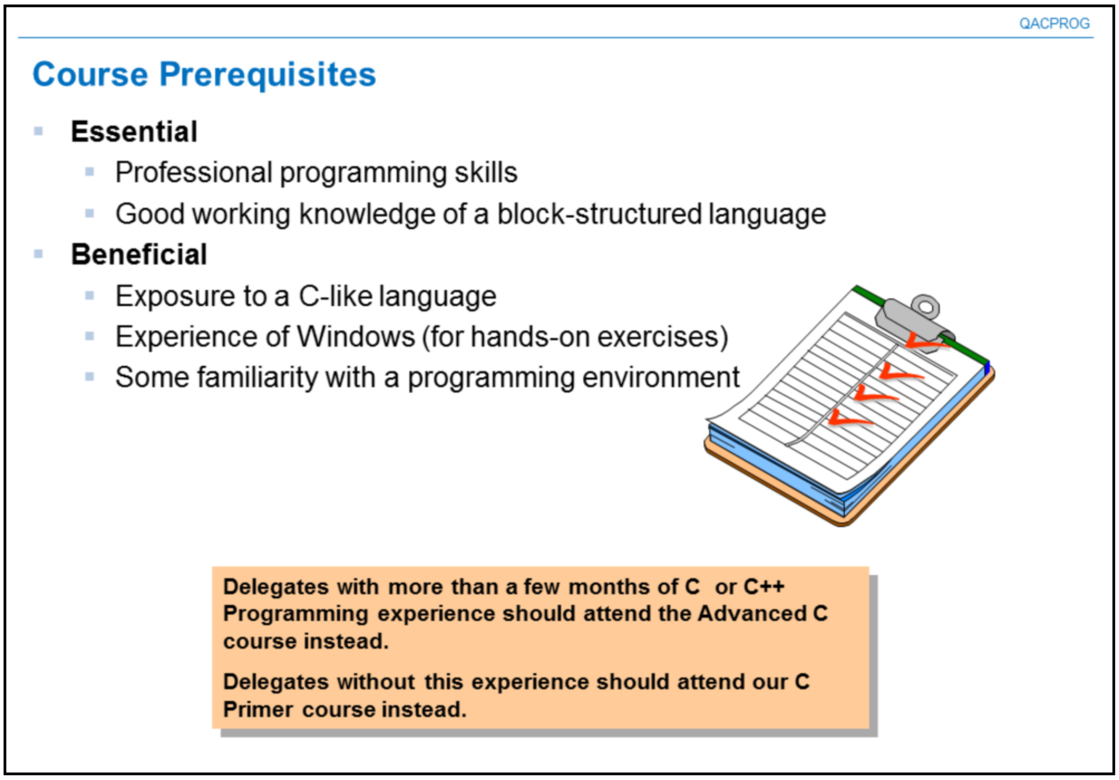


QACPROG

Course Introduction

- **Objective**
  - To explain the aims and objectives, training methods and C's position in the industry
- **Contents**
  - Course Prerequisites
  - Course Objectives
  - Course Delivery
  - Course Practicals
- **Summary**

The objectives for this chapter are to introduce the course structure, to provide an overall picture of the course content and format.



QACPROG

Course Prerequisites

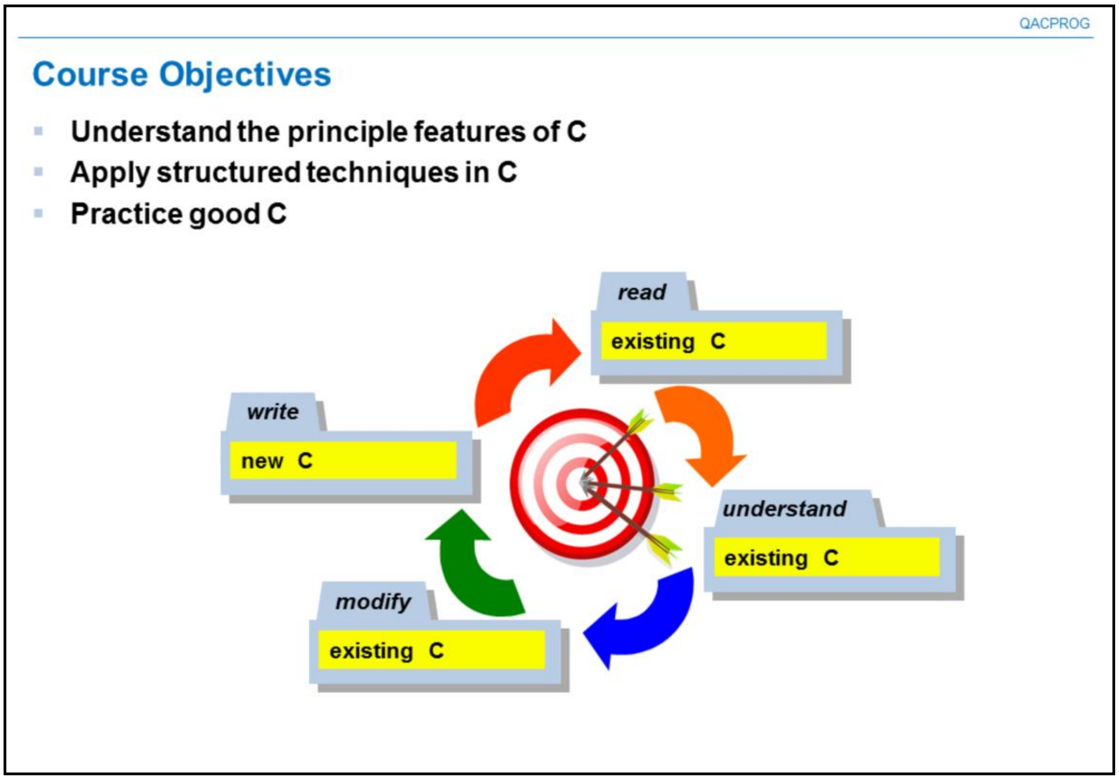
- **Essential**
  - Professional programming skills
  - Good working knowledge of a block-structured language
- **Beneficial**
  - Exposure to a C-like language
  - Experience of Windows (for hands-on exercises)
  - Some familiarity with a programming environment

Delegates with more than a few months of C or C++ Programming experience should attend the Advanced C course instead.

Delegates without this experience should attend our C Primer course instead.

If you feel that you are on the wrong course, tell us now! We will try to help you find the right one.

We do make some assumptions about you, as stated in the course outline and prerequisites in the course catalogue. And this slide!



QACPROG

Course Objectives

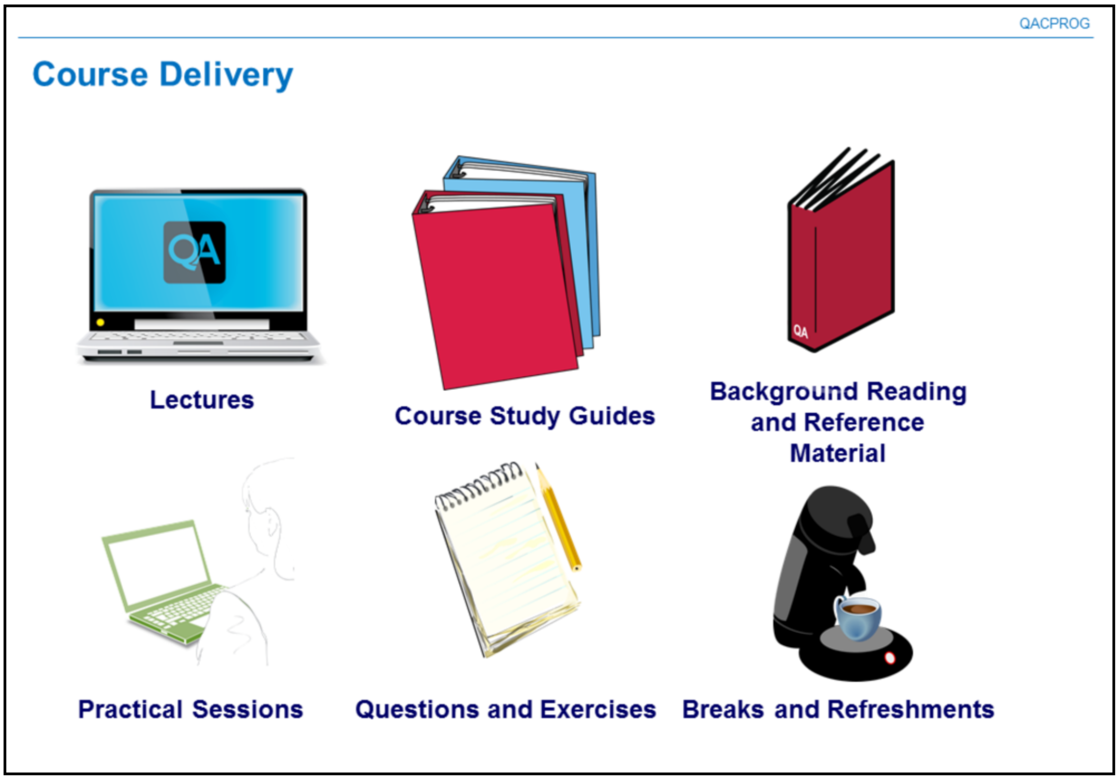
- Understand the principle features of C
- Apply structured techniques in C
- Practice good C

read
existing C
understand
existing C
modify
existing C
write
new C

The course has been designed with these criteria in mind. It is assumed that you are an experienced and proficient programmer without any or with a modest amount of C. This course is not an introduction to programming or an overview of C.
For many features in the language you can consider four levels of knowledge and experience:

- The entry level is to be able to *read* code that uses certain language features and syntax.
- The next level is to be able to *use* code that has already been written, for instance prewritten libraries.
- Then comes the ability to *modify* existing code, to change (or correct!) the way it behaves.
- The final level is to *write* new functions, classes and programs from scratch.

C is a small language but it can take some time to fully master and understand. Keep this classification in mind as you progress through the course.



Course Delivery

The course time will be divided between lecture periods and practical exercise sessions. The idea behind this is "listen and learn, see and understand, do and remember."

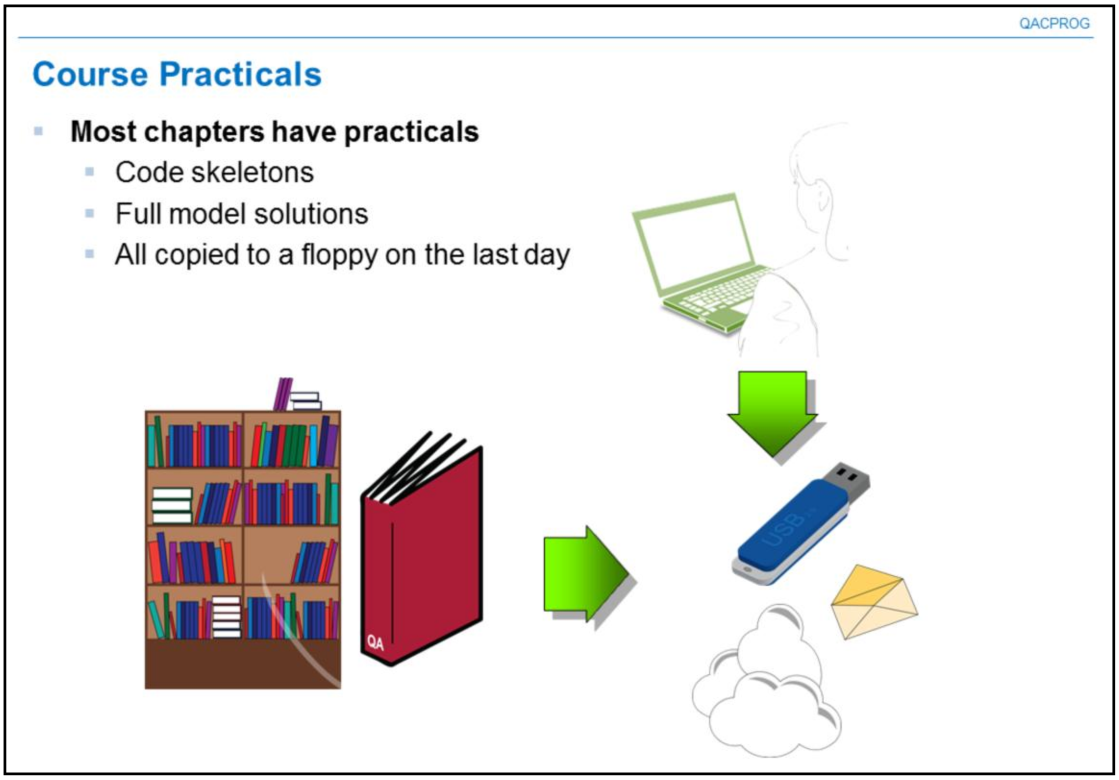
The lecture material includes many examples. The appendices provide reference and background reading material.

The course notebooks contain all the overhead foils that will be shown, so you do not need to copy them. In addition, there are extra textual comments (like these) below the foils, which should amplify the foils or provide further information.

The best courses are not those in which the instructor spends all his or her time pontificating at the front of the class. Things get more interesting if there is dialogue, so please feel free to make comments or ask questions. At the same time, the instructor has to think of the whole group, so if you have many queries, he or she may ask to deal with them off-line.

Work with other people during practical exercise sessions. The person next to you may have the answer, or you may know the remedy for the problem that your neighbour is having. The instructor is not the sole provider of answers; indeed, he or she reserves the right not to know everything and on occasions may have to find an answer and come back to you.

We are also individuals. We work at different paces and may have special interests in particular topics. The aim of the course is to provide a broad picture for all. Do not be dismayed if you do not appear to complete exercises as fast as the next person. The practical exercises are there to give plenty of practical opportunities; they do not have to be finished and you may even choose to focus for a long period on the topic that most interests you. If you have finished early, there is a great deal to investigate. Such "hacking" time is valuable. You may not get the opportunity to do it back in the office!

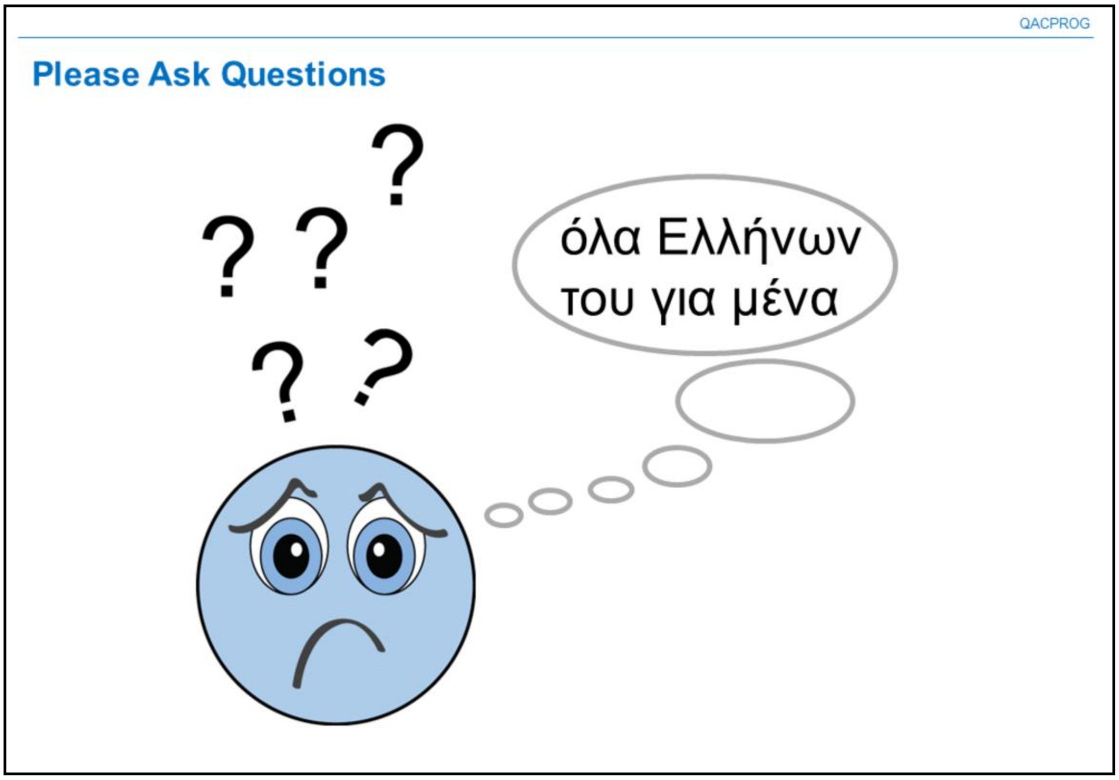

QACPROG

Course Practicals

- **Most chapters have practicals**
  - Code skeletons
  - Full model solutions
  - All copied to a floppy on the last day

The practical questions for each chapter are arranged so that they get progressively harder. Opening questions are based on the core material of the chapter, with later optional questions being based on more advanced material or detail. Many of the practicals in a session build on each other.

At the end of the course you can take away a copy of the all the practicals, the worked solutions, and the original code skeletons.



Please feel free to ask any questions at any time!

The importance of asking questions cannot be overstated. Do not be afraid to ask questions. Later chapters build on earlier chapters, so it is especially important to ask questions early on. There is no such thing as a stupid question.

Chinese proverb: "He who asks is a fool for five minutes, but he who does not ask remains a fool forever."

"To learn we must be willing to make mistakes." (Gerald Weinberg; The Psychology of Computer Programming, Dorset House, 0-932-633420)

Just in case you need a hint. όλα Ελλήνων του για μένα really means "It's all Greek to me" in Greek, naturally ☺