LONDON CAPITAL COMPUTER COLLEGE

Advanced Diploma in Computer Science (907) – Software Engineering

Prerequisites: Good computing knowledge	**Corequisites:** A pass or better in Diploma in System Analysis & Design or equivalence.
Aim: The principal objective of this course is produce candidates well versed in the principles of design, measurement and analysis, applied in the context of the development of software-based systems. A process used for building software is just like processes used for building any custom made product, hence a software engineer needs to follow a define engineering process to build software, just like an architect who constructs a building following a defined process. Various system development models, including current software engineering theory and practice, methodologies, techniques CASE tools are covered. Candidates also receive a solid foundation in technical communication skills, professional responsibility, social-effects, ethical aspects of software engineering practice, interpersonal relationships, teamwork and time management.	
Required Materials: Recommended learning resources.	**Supplementary Materials:** Lecture notes and tutor extra reading recommendations.
Special Requirements: Thorough research on software engineering.	
Intended Learning Outcomes:	**Assessment criteria:**
1 Define software engineering. Describe the historical origins of software engineering. Illustrate many activities of software engineering in addition to programming.	1.1 Analyse software engineering trend 1.2 Define software engineering 1.3 Define reasons why software engineering was born and the issues it addresses. 1.4 Describe relationship between software engineering and systems engineering. 1.5 Describe relationship between software engineering and programming. 1.6 Define main activities in software engineering. 1.7 Describe the meaning and importance of the software process.
2 Describe software – its nature and qualities. Analyse the classification of software qualities.	2.1 Analyse reasons for the human-intensiveness of software engineering. 2.2 Describe the meaning of functional requirements and the definition and kinds of qualities required from software 2.3 Difference between external and internal software qualities 2.4 Identify examples of external and internal software qualities 2.5 Identify examples of qualities required of software processes
3 Describe software engineering principles. Define the requirement process.	3.1 Define the important principles that form the basis of methods, techniques, methodologies and tools 3.2 Define the role of methodologies that package tools and techniques to encourage a particular approach to software development. 3.3 Describe the definitions and important principles used in phases of software development and in designing software processes

Tel: 0044 7423211037
Email: info@londoncomputercollege.co.uk Website: www.londoncomputercollege.co.uk
Registered No: 3267009 (England)

	3.4 Describe the importance of modularity as the cornerstone principle supporting software design. 3.5 Describe factors which leads to project failures 3.6 Define the process of capturing user requirements 3.7 Be able to describe modelling notations 3.8 Describe the requirements and specification languages 3.9 Define prototyping
4 Discuss the importance of design. Describe the architecture of a software system in terms of its components and their relationships. Describe how to use modularisation techniques.	4.1 Define program design. Describe the issues, techniques and characteristics of design 4.2 Define conceptual and technical design 4.3 Describe the different design styles, techniques and tools 4.4 Describe good design characteristics
5 Discuss the term “specification”. Describe the role and importance of specification in the different phases of software engineering.	5.1 Define a project schedule 5.2 Devise means of understanding customer needs 5.3 Define roles played by different personnel 5.4 Define the different types of costs involved 5.5 Illustrate risk management activities 5.6 Define how to track project progress 5.7 Describe project personnel. 5.8 Define risks management and illustrate the importance of a project plan.
6 Define the goals of verification. Describe the main approaches to verification. Describe the differences between formal and informal analysis and when to use each.	6.1 Describe testing requirements for concurrent and real-time systems. 6.2 Describe how to apply testing principles for object-oriented software. 6.3 Analyse the use of both formal and informal analysis techniques. 6.4 Describe the basic techniques of code inspections and walkthroughs. 6.5 Understand how to approach debugging systematically. 6.6 Describe the problems involved in verifying software qualities other than functional correctness and performance. 6.7 Analyse some of the metrics used to measure complexity, reliability, and performance. 6.8 Illustrate why traditional statistical models that are effective for measuring reliability in traditional engineering fields are difficult to apply to software. 6.9 Describe program testing process. Analyse the objective of software testing. 6.10 Describe why software fail 6.11 Describe the different types of software faults 6.12 Discuss who should perform software tests
7 Describe the phases of the traditional software life cycle. Identify the goals of software	7.1 Define software process models. Describe tools and techniques for

processes.	process modelling. 7.2 Analyse software development products, processes and resources 7.3 Understand the several models of the software development process 7.4 Describe the characteristics and limitations of the waterfall software process model. 7.5 Illustrate how to apply well-known methodologies: structured-analysis/structured-design (SA/SD) and Jackson system development method (JSD). 7.6 Describe the basic principles and phases of the unified process. 7.7 Analyse the importance and role of configuration management in the software life cycle. 7.8 Define the problems of legacy software and the processes and tools that focus on the activities in the maintenance of legacy software.
8 Describe problems encountered in managing software engineering projects. Define the key tasks of a project manager and challenges they face. Describe how productivity can be measured and the tools used for planning and monitoring.	8.1 Describe programming standards, procedures and guidelines 8.2 Define control structures 8.3 Evaluate the use of algorithms and data structures 8.4 Evaluate the importance of program re-use 8.5 Describe the problems inherent in organizing, controlling, and measuring intellectual activities. 8.6 Describe the common methods for measuring software productivity (lines of code and function points), and their limitations. 8.7 Describe the tools that managers use to plan and monitor projects. 8.8 Describe how to apply Work Breakdown Structures, GANTT and PERT charts in project management. 8.9 Describe typical and effective structures for organizing members of a team and their limitations and strengths. 8.10 Describe the capability maturity model for measuring the effectiveness of software organisations.
9 Describe the role and uses of CASE tools in software engineering.	9.1 Define objects. What is Object-Orientation (OO) 9.2 Define OO characteristics 9.3 Define objects and classes 9.4 Describe the OO development process 9.5 Define CASES 9.6 Define the role of editors, linkers, generators, interpreters, debuggers, analyzers, tracking tools, reverse engineering tools, and management tools in different phases of the software lifecycle. 9.7 Analyse the reasons and directions for the evolution of CASE tools.

10 Describe the impacts of software engineering on society. Describe ethical issues raised by software engineering.	10.1 Describe the principles of ethics adopted to guide software engineers in making ethical decisions 10.2 Analyse the influence of the Internet on software engineering 10.3 Describe the role of software as an enabling technology and the role of the Internet in providing new possibilities for software engineering.

Recommended Learning Resources: Software Engineering

Text Books	• Software Engineering: International Edition, 3/E by Shari Lawrence Pfleeger Joanne M Atlee ISBN-10: 0131984616 • Software Engineering: (Update), 8/E Ian Sommerville, *University of St. Andrews, United Kingdom* ISBN-10: 0321313798
Study Manuals	BCE produced study packs
CD ROM	Power-point slides
Software	None