

INTRODUCTION

Nineteen engineering disciplines are included in the Examination Syllabus issued by the Canadian Engineering Qualifications Board of Engineers Canada.

Each discipline examination syllabus is divided into two examination categories: compulsory and elective. A full set of Software Engineering examinations consists of ten, three-hour examination papers. Candidates will be assigned examinations based on an assessment of their academic background. Examinations from discipline syllabi other than those specific to the candidates' discipline may be assigned at the discretion of the constituent Association/Ordre.

Before writing the discipline examinations, candidates must have passed, or have been exempted from, the Basic Studies Examinations.

The constituent Association/Ordre will supply information on examination scheduling, textbooks, materials provided or required, and whether the examinations are open or closed book.

SOFTWARE ENGINEERING EXAMINATIONS

GROUP A

COMPULSORY EXAMINATIONS (SEVEN REQUIRED)

04-Soft-A1 Algorithms & Data Structures

Fundamental data structures and their associated algorithms. Stacks and queues, trees, tables, lists, arrays, strings, sets; files and access methods. B-trees, multi-key organizations. Searching. Sorting.

Algorithm design techniques, such as divide and conquer, the greedy method, balancing, dynamic programming. Algorithms related to set operations, Graphs, graph algorithms: depth-first and breadth-first search, minimum spanning tree, shortest path. Empirical and theoretical measures of the efficiency of algorithms.

Complexity analysis. Hard problems, NP-completeness, and intractable problems

04-Soft-A2 Computing Structures

Computer Architecture basics, including Boolean algebra, gates, combinational and sequential logic, machine-level representation of data; machine organization, assembly/machine language programming; memory organization, caches, heaps, stacks; serial and parallel I/O, interrupts, bus protocols, and direct-memory access (DMA). Operating System basics, including concurrency, process scheduling, memory management; protection, access, and authentication; linking and loading. Database basics, including database architecture, query languages, transactions. File system organization and access methods (sequential, indexed-sequential, extendible hashing, B-trees), index organization.

04-Soft-A3 Software Design

Role of software design activity. Software design quality attributes (correctness, reliability, maintainability, portability, robustness). Software design principles (separation of concerns, abstraction, information hiding). Software architecture, architectural structures and views. Modularity and decomposition. Function-oriented design. Object-oriented design. Components. Interface design. Module level design. Specification for design. Notations (graphical and languages). UML. Basic concepts of design patterns.

04-Soft-A4 Real Time Systems

Definition and characteristics of real-time systems. Hard and soft real-time systems. Dynamic responses of simple physical processes. Designing real-time systems (requirements, design methods, implementation, testing, human-computer interaction). Reliability and fault tolerance. Exceptions and exception handling. Concurrency. Synchronization and communication. Resource control. Scheduling (cyclic executive, rate monotonic and deadline priority, priority ceiling protocols). Real-time operating systems. Simple embedded systems.

04-Soft-A5 Requirements and Specifications

Elicitation sources and techniques. Modelling paradigms, including information modelling, behavioural modelling, domain modelling, functional modelling, constraint modelling. Quality requirements (e.g., performance, usability, reliability, maintainability); expressing quality requirements so that they are testable. Prioritization, trade-off analysis, negotiation, risk analysis, and impact analysis. Requirements management, consistency management, interaction analysis, traceability. Requirements documentation and specification languages. Validation, reviews and inspections, prototyping, validating non-functional requirements. Acceptance test design.

04-Soft-A6 Software Quality Assurance

Validation and Verification concepts, Software Lifecycle and application of validation and verification, Software Quality Assurance processes, Definitions of software product quality, Quality Characteristics, Engineering quality definitions, specifications, Definition and classifications of software Defects, Fitness for use and customer quality definitions, Software Costs, quality costs and economics, Reviews, Walkthroughs and Inspections: General Concepts, Unit (Module / Package) level testing, Subsystem / Integration testing, Regression testing, State based testing, Traditional Functional Testing, Logical Testing/Analysis, OO Testing considerations (polymorphism and inheritance), Safety / Failure Analysis and testing.

04-Soft-A7 Software Process

Software life cycles. Software process models. Activities in each phase. Control and life-cycle management of correct, reliable, maintainable and cost effective software. Software documentation. Project management tools. Risk management. Communication and collaboration. Cause and effects of project failure. Cost estimation and scheduling. Factors influencing productivity and success. Productivity metrics. Planning for change. Managing expectations. Software maintenance. Configuration management.

GROUP B

ELECTIVE EXAMINATIONS (THREE REQUIRED)

04-Soft-B1 Advanced Object Oriented Design

Object orientation paradigm. Object oriented analysis: basic concepts, use cases, analysis, stereotypes and objects, analysis patterns. Object modelling languages. Object oriented design: basic concepts, design stereotypes and objects, design patterns. Object oriented programming: basic concepts, idioms, most common object oriented programming languages(C++, Java). Application frameworks. Object oriented case tools. State transition and interaction diagrams. Testing of object oriented programs.

Advanced approaches to object-oriented analysis and design. Frameworks and design patterns. Design for reusability. Advanced object-oriented programming techniques. Design using object-oriented databases and distributed object architectures. Design of software agents. Project involving object-oriented analysis, design, and implementation.

04-Soft-B2 User interface

Psychological principles of human-computer interaction. Evaluation of user interfaces. Usability engineering. Task analysis, user-centered design and prototyping. Conceptual models and metaphors. Software design rationale. Design of windows, menus and commands. Voice and natural language I/O. Response time and feedback. Colour, icons and sound. Internationalization and localization. User interface architectures and APIs. Case studies and project.

04-Soft-B3 Security/Safety

Characteristics of security problems. Basic encryption and decryption; secure encryption, encryption protocols and practices. Public key cryptosystems and digital signatures. Software design for security. Security in operating systems, databases, personal computers, computer networks and electronic communications. Risk analysis and security planning. Ethical issues in computer security.

04-Soft-B4 Reliability and Fault Tolerance

Software and hardware faults. Faults, latent faults and failures. Characterization of failure functions, probability distribution of failures, failure intensity function. Software reliability definition and measures. MTTF, MTBF, MTTR, availability, maintainability. System use, Musa's operational profiles and type-1 uncertainty. Defect removal and type-2 uncertainty. Reliability stability and reliability growth. Distinction between hardware reliability and software reliability. Failure probability density function and reliability function. Systems of reliability prediction, including prediction models. Statistical testing. Characterization of fault Injection, detection and correction at various stages within the lifecycle. Techniques for prediction of remaining faults, including fault injection, classification tree analysis, code coverage. General lifecycle measures for producing reliable software, including defect prevention; early defect detection and removal; design for robustness; use of measurements from V&V activities; stabilization of requirements, design, code and test artifacts. Active and Passive fault detection. N-version programming. Fault handling and correction, exceptions, fault tolerance. FTA. Survivability, critical functions and degraded modes of operation.

04-Soft-B5 Software Modeling & Verification (Formal Methods)

Mathematical modelling of software, including logic, extended finite state machines, process algebra, functions, and algebraic specifications. Mathematical reasoning of such models, including proofs of consistency, completeness, and correctness. Tools for type checking, well-formedness checking, simulation, invariant and property checking (e.g., deadlock checking, model checking), test-case generation, and code generation.

04-Soft-B6 Advanced Software Project Management, Life Cycle Methodologies

Software Project Management Processes

Planning, integration and change control; scope management; quality management; cost management; risk management; schedule management; communications management; human resource management; and procurement management.

Lifecycle Methodologies

Development models (Waterfall, V-Model, Incremental, Spiral, etc.) and techniques (rapid prototype, clean room, Object Oriented, etc.) Military/aerospace and commercial development standards. Phase specific activities: software specification and requirements analysis; software architecture comparison and selection; software design; coding and unit testing; integration testing; system testing; and operational turn-over. Lifecycle activities: baseline management, software quality management, software configuration management.

Project involving forensic project management of a real industrial project.

04-Soft-B7 Reverse Engineering, Maintenance & Evolution

Software maintenance: corrective, perfective, and adaptive. Techniques for reverse engineering software architecture and design, for the purpose of program comprehension. System and process re-engineering (technical and business). Refactoring. Migration (technical and business). Impact analysis. Release and configuration management. Models of software evolution (theories, laws). Relationship among evolving entities (e.g., assumptions, requirements, architecture, design, code, test suites). Legacy systems.

04-Soft-B8 Distributed Systems

Characteristics of distributed systems. Networked vs. centralized systems. Fundamental concepts and mechanisms. Architectural concepts of distributing an application over several platforms. Overview of network configurations and topologies. Client-server systems. Process synchronization and inter-process communications. Principles of fault tolerance. Transaction processing techniques. Distributed file systems. Operating systems for distributed architectures. Security.

04-Soft-B9 Parallel Computing

Models of parallel computation. Superscalar architecture. Shared memory parallel machines. Interconnection networks and their topological properties. Massively parallel computers. Hypercube architectures. Performance measurement for parallel algorithms. Parallel evaluation of expressions. Parallel searching and data structures. Parallel algebraic and geometric processing.

04-Soft-B10 Networking and Communications

Data communications, including signals, modulation, and reception. Data transmission protocols, including half/full duplex, asynchronous/synchronous, point-to-point/multidrop, and character/bit oriented. Error detecting and correcting codes. Character sets, switching alternatives, including circuit, message, and packet. The OSI model, including physical, data link, and network layer protocols (RS 232C, RS 449-422/23, HDLC, X.25). Transport protocols. Session layer and ISDN. The higher layers. Introduction to network structures and designs. Switching techniques. Resource access/sharing methods (random access, polling, concentration and multiplexing). Public data networks. Local area networks, including bus, ring, and tree topologies, protocols, and hardware. Metropolitan and Wide area networks. Elements of queuing systems. Network performance measures (queue length, delay and throughput). Standards and the standardization process.

04-Soft-B11 Process Control Systems

Continuous and discrete time models of continuous physical phenomena. Laplace transform, Z-transform, transfer functions. Time domain and frequency domain response of first, second and higher order systems. Stability and feedback compensation. Steady state error and proportional, integral and derivative (PID) control. Compensator design using root locus analysis, Nyquist criterion, and frequency domain design. Sampling theorem, aliasing and digital implementation of continuous controllers. Anti-alias pre-filtering. Software implementation of digital compensators. Computer control interfacing.

04-Soft-B12 Scientific Computation

Representation of numbers and floating point round-off. Polynomials: representation, division. Zeros of functions: bisection, Newton-Raphson. Numerical linear algebra: LU decomposition and Gauss elimination, conditioning, iterative techniques, structured systems, eigenvalues. Numerical differentiation. Quadrature: Simpson's rule, Gauss quadrature. Interpolation: Lagrange, splines. Data smoothing: least squares. Ordinary differential equations: initial value problems, predictor-corrector, Runge-Kutta, boundary value problems, systems of ODEs. Simple partial differential equations: parabolic, hyperbolic and elliptic problems.

04-Soft-B13 Performance Analysis & Simulation

Basic techniques of system performance evaluation. Specific topics include: performance modeling, discrete event simulation, verification and validation of simulation models, analysis of simulation output, analysis of single server queue and queuing networks, modeling of computer systems, networks, and other queuing or non-queuing systems.

04-Soft-B14 Safety Critical Systems

Safety and hazard analysis. Use of software in safety related systems. Legal and ethical considerations. FMEA, HAZOP, FTA, ETA. Risk assessment – frequency and consequences of hazardous events. Safety integrity levels and safety cases. Software reliability. Distinction between safety and reliability of systems. Achievement of software reliability by fault prevention and fault tolerance. Software design aspects for safety and fault tolerance – including HCI. Choice of programming language, safe subsets. Formal methods, Algebraic, model and process based specification, formal specification languages, refinement proofs, verification proofs. Fault tolerance, redundancy and common mode failures, N-version programming and recovery blocks. Safety related standards. Certification and safety cases.

04-Soft-B15 Artificial Intelligence/Intelligent Systems

Artificial intelligence; definition and applications. Concepts of artificial intelligence. Predicate calculus and its use in artificial intelligence Overview of knowledge-based and expert systems. Production systems. Knowledge representations. Search methods; planning. Natural language processing. Programming languages (LISP and Prolog) for AI and expert system implementation. Knowledge representation. Rule-based and object-based systems. Vision.

04-Soft-B16 Compilers

Fundamental features of programming languages, functions of compilers. Processors, preprocessors, linkers and translators. Organization of compilers including compile-time and run-time tables. Lexical and syntactical analysis. Intermediate code generation. Object code generation. Error diagnostic. Code optimization.

04-Soft-B17 Programming Language Paradigm

Principles of programming methodologies and the evolution of programming-language features (e.g., modules, objects, inheritance, polymorphism, exceptions, templates) to support those methodologies. Examination of major programming-language paradigms, including procedural, logic, functional, and object-oriented. Programming knowledge in a variety of languages.

04-Soft-B18 Computer Graphics/Imaging/Visualization

Graphics hardware, architecture, and devices. Vector and raster graphics algorithms and systems. Techniques for describing and generating images: point, vector, and raster approaches. Image transformation, including scaling, translation, rotation, clipping, and windowing. Graphic software and data structures. Geometrical transformations. Viewing in three dimensions. Curves and splines. Picture generation using solid polyhedra. Illumination and color. Ray tracing. Display file compilers. Graphics data structures. Interactive graphics. Graphic standards such as GKS, PHIGS, TIGA, and X-WINDOWS. Virtual reality technology.

Image acquisition. Image thresholding, edge detection techniques. Contour followers. Image filtering. Mathematical morphology. Image segmentation. Pattern recognition techniques. Matching techniques. 3D object reconstruction. Image compression and related standards.