

KOENIG SOLUTIONS • COURSE OUTLINE

Computer System and Algorithm Programming

Learn to explain computer systems architecture, networks and security, and design, write and test algorithms and programs.

32 Hours

DURATION

9

MODULES

169

TOPICS

COURSE OVERVIEW

This course covers the architecture and performance of the CPU, primary and secondary storage, data representation and compression, and computer networks, protocols and security. It then addresses operating systems and utility software before moving into computational thinking, algorithm design, searching and sorting. It continues with programming fundamentals, data types and techniques such as string manipulation, file handling, SQL and arrays, followed by defensive design and testing practices. It closes with a look at programming languages, translators and integrated development environments.

WHO THIS COURSE IS FOR

Computer Science Students

Software Developers

Systems Analysts

Network Administrators

IT Support Staff

LEARNING OBJECTIVES

- Explain the architecture and performance factors of the CPU
- Describe primary and secondary storage and units of data
- Interpret binary, hexadecimal and character, image and sound data
- Identify network topologies, hardware, protocols and layers
- Assess common threats and vulnerabilities to computer networks
- Apply computational thinking to design and refine algorithms
- Implement programs using variables, data types and control constructs
- Apply defensive design and testing techniques to produce robust programs

9 Modules, 169 Topics

1.1**2.5 hours**

Systems Architecture

1.1.1 ARCHITECTURE OF THE CPU

- **The purpose of the CPU:** The fetch-execute cycle
- **Common CPU components and their function:** ALU (Arithmetic Logic Unit)
- CU (Control Unit)
- Cache
- Registers
- **Von Neumann architecture:** MAR (Memory Address Register)
- MDR (Memory Data Register)
- Program Counter
- Accumulator

1.1.2 CPU PERFORMANCE

- **How common characteristics of CPUs affect their performance:** Clock speed
- Cache size
- Number of cores

Memory and Storage

1.2.1 PRIMARY STORAGE (MEMORY)

- The need for primary storage
- The difference between RAM and ROM
- The purpose of ROM in a computer system
- The purpose of RAM in a computer system
- Virtual memory
- Cache

1.2.2 SECONDARY STORAGE

- The need for secondary storage
- Common types of storage: Optical
- Magnetic
- Solid state
- Suitable storage devices and storage media for a given application
- The advantages and disadvantages of different storage devices and storage media relating to these characteristics: Capacity
- Speed
- Portability
- Durability
- Reliability
- Cost

1.2.3 UNITS

- The units of data storage: Bit
- Nibble (4 bits)
- Byte (8 bits)
- Kilobyte (1,000 bytes or 1 KB)
- Megabyte (1,000 KB)
- Gigabyte (1,000 MB)
- Terabyte (1,000 GB)
- Petabyte (1,000 TB)
- How data needs to be converted into a binary format to be processed by a computer
- Data capacity and calculation of data capacity requirements

1.2.4 DATA STORAGE

- Numbers
- How to convert positive denary whole numbers to binary numbers (up to and including 8 bits) and vice versa
- How to add two binary integers together (up to and including 8 bits) and explain overflow errors which may occur
- How to convert positive denary whole numbers into 2-digit hexadecimal numbers and vice versa
- How to convert binary integers to their hexadecimal equivalents and vice versa
- Binary shifts
- Characters
- The use of binary codes to represent characters
- The term 'character set'
- The relationship between the number of bits per character in a character set, and the number of characters which can be represented, e.g.: ASCII
- Unicode
- Images
- How an image is represented as a series of pixels, represented in binary
- Metadata
- **The effect of colour depth and resolution on:** The quality of the image
- The size of an image file
- Sound
- How sound can be sampled and stored in digital form
- The effect of sample rate, duration and bit depth on: The playback quality
- The size of a sound file
- **1.2.5 COMPRESSION**
- The need for compression
- Types of compression: Lossy
- Lossless

Computer Networks, Connections and Protocols

1.3.1 NETWORKS AND TOPOLOGIES

- **Types of network:** LAN (Local Area Network)
- WAN (Wide Area Network)
- Factors that affect the performance of networks
- The different roles of computers in a client-server and a peer-to-peer network
- The hardware needed to connect stand-alone computers into a Local Area Network: Wireless access points
- Routers
- Switches
- NIC (Network Interface Controller/Card)
- Transmission media
- **The Internet as a worldwide collection of computer networks:** DNS (Domain Name Server)
- Hosting
- The Cloud
- Web servers and clients
- Star and Mesh network topologies

1.3.2 WIRED AND WIRELESS NETWORKS, PROTOCOLS AND LAYERS

- **Modes of connection:** Wired: Ethernet
- Wireless: Wi-Fi
- Wireless: Bluetooth
- Encryption
- IP addressing and MAC addressing
- Standards
- **Common protocols including:** TCP/IP (Transmission Control Protocol/Internet Protocol)
- HTTP (Hyper Text Transfer Protocol)
- HTTPS (Hyper Text Transfer Protocol Secure)
- FTP (File Transfer Protocol)
- POP (Post Office Protocol)
- IMAP (Internet Message Access Protocol)
- SMTP (Simple Mail Transfer Protocol)
- The concept of layers

1.4**2 hours**

Network Security

1.4.1 THREATS TO COMPUTER SYSTEMS AND NETWORKS

- Forms of attack: Malware
- Social engineering, e.g. phishing, people as the 'weak point'
- Brute-force attacks
- Denial of service attacks
- Data interception and theft
- The concept of SQL injection

1.4.2 IDENTIFYING AND PREVENTING VULNERABILITIES

- **Common prevention methods:** Penetration testing
- Anti-malware software
- Firewalls
- User access levels
- Passwords
- Encryption
- Physical security

1.5**2 hours**

Systems Software

1.5.1 OPERATING SYSTEMS

- **The purpose and functionality of operating systems:** User interface
- Memory management and multitasking
- Peripheral management and drivers
- User management
- File management

1.5.2 UTILITY SOFTWARE

- The purpose and functionality of utility software
- **Utility system software:** Encryption software
- Defragmentation
- Data compression

2.1

4 hours

Algorithms

2.1.1 COMPUTATIONAL THINKING

- Principles of computational thinking: Abstraction
- Decomposition
- Algorithmic thinking

2.1.2 DESIGNING, CREATING AND REFINING ALGORITHMS

- Identify the inputs, processes, and outputs for a problem
- Structure diagrams
- Create, interpret, correct, complete, and refine algorithms using: Pseudocode
- Flowcharts
- Reference language/high-level programming language
- Identify common errors
- Trace tables

2.1.3 SEARCHING AND SORTING ALGORITHMS

- **Standard searching algorithms:** Binary search
- Linear search
- **Standard sorting algorithms:** Bubble sort
- Merge sort
- Insertion sort

2.2

10 hours

Programming Fundamentals

2.2.1 PROGRAMMING FUNDAMENTALS

- The use of variables, constants, operators, inputs, outputs and assignments
- The use of the three basic programming constructs used to control the flow of a program: Sequence
- Selection
- Iteration (count- and condition-controlled loops)
- The common arithmetic operators
- The common Boolean operators AND, OR and NOT

2.2.2 DATA TYPES

- The use of data types: Integer
- Real
- Boolean
- Character and string
- Casting

2.2.3 ADDITIONAL PROGRAMMING TECHNIQUES

- The use of basic string manipulation
- The use of basic file handling operations: Open
- Read
- Write
- Close
- The use of records to store data
- The use of SQL to search for data
- The use of arrays (or equivalent) when solving problems, including both one-dimensional (1D) and two-dimensional arrays (2D)
- How to use sub programs (functions and procedures) to produce structured code
- Random number generation

2.3

3 hours

Producing Robust Programs

2.3.1 DEFENSIVE DESIGN

- Input validation
- **Maintainability:** Use of sub programs
- Naming conventions
- Indentation
- Commenting

2.3.2 TESTING

- The purpose of testing
- Types of testing: Iterative
- Final/terminal
- Identify syntax and logic errors
- Selecting and using suitable test data: Normal
- Boundary
- Invalid/Erroneous
- Refining algorithms

2.5

2 hours

Programming Languages and Integrated Development Environments

2.5.1 LANGUAGES

- **Characteristics and purpose of different levels of programming language:** High-level languages
- Low-level languages
- The purpose of translators
- The characteristics of a compiler and an interpreter

2.5.2 THE INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

- Common tools and facilities available in an Integrated Development Environment (IDE): Editors
- Error diagnostics
- Run-time environment
- Translators

WHAT IS INCLUDED

Everything You Need to Succeed

- Koenig course material for every module
- Delivery by certified instructors and subject matter experts
- Hands-on exercises throughout the course
- Course completion certificate from Koenig

ABOUT KOENIG

Who We Are

Established in 1993, Koenig Solutions is a globally recognised training provider with 35+ technology partnerships, 5,000+ training programmes and a network of certified instructors and subject matter experts, supporting individuals and enterprise teams worldwide.

FLEXIBLE DELIVERY

Learn the Way That Suits You

Classroom Training

Live Online Classes (ILO)

Fly-Me-A-Trainer (FMAT)

Flexi (Self-Paced)

1-on-1 Training

Ready to Get Started?

Talk to a learning advisor about schedules, group training and customised delivery.

Visit koenig-solutions.com

✉ info@koenig-solutions.com