COS1521 - Computer Systems: Fundamental Concepts

Summary of Textbook Chapters

Foundations of Computer Science 2nd ed. - Forouzan & Mosharraf

Notes:

This summary includes: Key Terms, and end of chapter Summaries.
It is designed to be printed back-to-back.
See Tutorial Letter 102 for examples of calculations

Ron Barnard

Ver 1.0 Oct 2012

Chapter 1 - Introduction

Key Terms

- **algorithm** - An ordered set of unambiguous steps that produces a result and terminates in a finite time.
- **arithmetic logic unit** (ALU) - Where calculation & logical operations take place.
- **computer languages –** Any of the syntactical languages used to write programs for computers, such as machine language, assembly language, C, COBOL and Fortran.
- **control unit** - Controls the operations of the memory, ALU, and input / output sub-system.
- **data processor –** An entity that inputs data, processes it, and outputs the result.
- **digital divide** – A social issue that divides people in society into two groups: those who are electronically connected to the rest of society, and those who are not.
- **input data** – User information that is submitted to a computer to run a program.
- **instruction** – A command that tells a computer what to do.
- **integrated circuit** – Transistors, wiring, and other components on a single chip.
- **memory** - Where programs and data are stored.
- **operating system** – The software that controls the computing environment and provides an interface to the user.
- **output data** – The results of running a computer program.
- **program** – A set of instructions.
- **software engineering** - The design & writing of computer programs, following strict rules & principles.
- **Turing machine** – A computer model with 3 components: tape, controller, and read/write head, that can implement statements in a computer language.
- **Turing model** – A computer model based on Alan Turings' theoretical definition of a computer.
- **von Neumann model** – A computer model consisting of memory, arithmetic logic unit, control unit, and input/output subsystems, upon which the modern computer is based.

Summary

The idea of a universal computational device was first given by Alan Turing in 1937. He proposed that all computation can be performed by a special kind of machine, now called a Turing machine.

The von Neumann model defines a computer as four subsystems: memory, arithmetic logic unit, control unit, and input / output. The von Neumann model states that the program must be stored in memory.
We can think of a computer as made up of three components: computer hardware, data, and computer software.

The history of computing and computers can be divided into three periods: the period of mechanical machines (before 1930), the period of electronic computers (1930 – 1950), and the period that includes the five modern computer generations:

- 1st 1950 -1959: Bulky, used vacuum tubes, affordable only by big organizations;
- 2nd 1959 -1965: Used transistors, affordable to small & medium sized corporations, FORTRAN & COBOL were invented;
- 3rd 1965 - 1975: Integrated circuits reduced size & cost of computers. Software packages became available, new software industry was born;
- 4th 1975 -1985: Appearance of micro-computers. Emergence of computer networks;
- 5th 1985 -> Appearance of laptop and palmtop computers, improvements in storage media (CD-ROM, DVD), the use of multimedia and virtual reality.

Computer science has created some peripheral issues, the most prevalent of which can be categorized as social: (dependency, social justice, digital divide), and ethical issues: (privacy, copyright, computer crime).

With the invention of computers a new discipline has evolved, *computer science*, which is now divided into several areas.

--ooOoo--

Chapter 2 - Number Systems

Key Terms

- **base** – The number of symbols in a numbering system.
- **binary digit** (bit) – The smallest unit of information (0 or 1).
- **binary system** – A numbering system that uses two symbols: 0 and 1.
- **bit** – Acronym for binary digit.
- **decimal digit** – A symbol in the decimal system.
- **decimal system** – A method of representing numbers using ten symbols: 0 to 9.
- **hexadecimal digit** – A symbol in the hexadecimal system.
- **hexadecimal system** – A numbering system with base 16. Its digits are: 0 to 9, A, B, C, D, E, F.
- **integer** – An integral number, a number without a fractional part.
- **number system** – A system that uses a set of symbols to define a value.
- **octal digit** – A symbol in the octal system.
- **octal system** – A number system with base 8. Its digits are: 0 to 7.
- **place value** – The value related to a position in positional number system.
- **positional number system** – A number system in which the position of a symbol in a number defines its value.
- **radix** (base) – The base in a positional number system.
- **real** – A number with both integral and fractional parts.

Summary

A number system (or numeral system) is a system that uses distinct symbols to represent a number.

In a positional number system, the position a symbol occupies in the number determines the value it represents. Each position has a place value associated with it.

(Non-positional number system omitted)

In the decimal system, the base b = 10 and we use 10 symbols to represent numbers. The symbols in this system are often referred to as *decimal digits* or just *digits*.

In the binary system, the base b = 2 and we use only two symbols to represent numbers. The symbols in this system are often referred to as *binary digits* or *bits*.

In a hexadecimal system, the base b = 16 and we use sixteen symbols to represent numbers. The symbols in this system are often referred to as *hexadecimal digits*.

In an octal system, the base b = 8 and we use eight symbols to represent numbers. The symbols in this system are often referred to as *octal digits*.

We can convert a number in any system to decimal. We multiply each digit with its place value in the source system and add the result to get the number in the decimal system.

We can convert a decimal number to its equivalent in any base using two different procedures, one for the integral part and one for the fractional part. The integral part needs repeated division and the fraction part needs repeated multiplication.

Conversion from the binary system to the hexadecimal system and from the hexadecimal system to the binary system is very easy, because four bits in the binary system are represented as one digit n the hexadecimal system.

Conversion from the binary system to the octal system and from the octal system to the binary system is very easy, because three bits in the binary system are represented as one digit in the octal system.

--ooOoo--

Chapter 3 - Data Storage

Key Terms

- **American National Standards Institute (ANSI)** – An organization that creates standards in programming languages, electrical specifications, communication protocols etc.
- **American Standard Code for Information Interchange (ASCII)** – An encoding scheme that defines control and printable characters for 128 values.
- **analogue** – A continuously varying entity.
- **audio** – Recording or transmission of sound or music.
- **binary digit** – The smallest unit of information (0 or 1)
- **bit** – Acronym for *binary digit*.
- **bit depth** – The number of bits representing a sample in a sampling process.
- **bit pattern** – A sequence of bits (0's and 1's).
- **bit rate** – The number of bits transmitted per second.
- **bitmap graphic** – A graphic representation in which a combination of pixels defines the image.
- **byte** – A unit of storage, usually 8 bits.
- **code** – A set of bit patterns designed to represent text symbols.
- **color depth** – The number of bits used to represent the colour of a pixel.
- **digital** – A discrete (non-continuous) entity.
- **encoding** – The conversion of *quantized* sample values to bit patterns.
- **Excess representation** – n/a
- **Excess_1023** – n/a
- **Excess_127** – n/a
- **fixed-point representation** – Representation of numbers in a computer in which the position of the decimal point is fixed. Normally used to represent integers.
- **floating-point representation** – A number representation in which the position of the decimal point is floated to create better precision. Normally used to represent real numbers in a computer.
- **Graphic Interchange Format (GIF)** – An 8-bit per pixel bitmap image.
- **indexed color** – A technique in raster graphics that uses only a portion of True-Color to encode colours in each application.
- **MP3** – A standard used for compression of audio, based on MPEG.

- **MPEG** – A lossy compression method for compressing video and audio.
- **normalization** – In floating-point representation, to make the fixed part of the representation uniform.
- **one's complement** – A bitwise operation that reverses the value of the bits in a variable.
- **overflow** – The condition that results when there are insufficient bits to represent a number in binary.
- **quantization** – A process that rounds the value of a sample to the closest integer value.
- **raster graphic** – See *bitmap graphic.*
- **real** – A number with both integral and fractional parts.
- **resolution** – The scanning rate in image processing: the number of pixels per unit measure.
- **RGB** – A colour system in which tints are represented by a combination of red, green, and blue primary colours.
- **sampling** – Taking measurements at equal intervals.
- **sampling rate** – The number of samples obtained per second in the sampling process.
- **scanning** – Converting an image into digital data by sampling its density and colour at evenly-spaced points.
- **sign-and-magnitude representation** – A method of representing integers in which 1 bit represents the sign of the number, and the remaining bits represent the magnitude.
- **text** – Data stored as characters.
- **True-Color** – A technique in raster graphics that uses 24 bits to represent a colour.
- **truncation error** – An error that occurs when a number is stored using floating-point representation. The value of the number stored may not be exactly as expected.
- **two's complement** – A representation of binary numbers in which the complement of a number is found by complementing all bits and adding 1.
- **underflow** – An event that occurs when an attempt is made to delete data from an empty data structure.
- **Unicode** – A 32 bit code that includes the symbols and alphabets from most languages in the world.
- **unsigned integer** – An integer without a sign whose value ranges between 0 and positive infinity.
- **vector graphic** – A type of graphics file format in which lines and curves are defined using mathematical formulas.
- **video** – A representation of images, called frames, in time.
- **word** – A bit pattern longer than a byte.

Summary

Data comes in different forms, including numbers, text, audio, image, and video. All data types are transformed into a uniform representation called a *bit pattern*.

A number is changed to the binary system before being stored in computer memory. There are several ways to handle the sign. There are two ways to handle the decimal point: fixed-point and floating-point.

An integer can be thought of as a number in which the position of the decimal point is fixed: the decimal point is at the right of the least significant bit. An unsigned integer is an integer that can never be negative.

One of the methods used to store a signed integer is the sign-and-magnitude format. In this format, the leftmost bit is used to show the sign and the rest of the bits define the magnitude. Sign and magnitude are separated from each other.

Almost all computers use the *two's complement representation* to store a signed integer in an n-bit memory location. In this method, the available range for unsigned integers is divided into two equal subranges. The first half is used to represent non-negative integers, the second half is used to represent negative integers. In two's complement representation, the leftmost bit defines the sign of the integer, but sign and magnitude are not separated from each other.

A *real* is a number with an integral part and a fractional part. Real numbers are stored in the computer using *floating-point representation*. In floating-point representation a number is made up of three sections: a sign, a shifter, and a fixed-point number.

A piece of text in any language is a sequence of symbols. we can represent each symbol with a bit pattern. Different sets of bit patterns (codes) have been designed to represent text symbols. A coalition of hardware and software manufacturers have designed a code called *Unicode* that uses 32 bits to represent a symbol.

Audio is a representation of sound or music. Audio is analogue data. we cannot record an infinite number of values in an interval, we can only record some samples. The number of samples depends on the maximum number of changes in the analogue signal. The values measured for each sample is a real number. *Quantization* refers to a process that rounds up the sample values to integers.

Storage of images is done using two different techniques: *raster graphics* and *vector graphics*. Raster graphics are used when we need to store an analogue image such as a photograph. The image is scanned (sampled) and pixels are stored. In the vector graphic method, an image is decomposed into a combination of geometrical shapes, such as lines, squares, or circles. Each geometrical shape is represented by a mathematical formula.

Video is a representation of images, called *frames,* in time. A movie is a series of frames shown one after another to create the illusion of continuous motion. In other words, video is the representation of information that changes in space (single image) and in time (a series of images).

--ooOoo--

Chapter 4 - Operations on Data

Key Terms

- **AND operation** – A bit level operation: the result of the operation is 1 only if both bits are 1, otherwise 0.
- **arithmetic operation** – An operation that takes 2 numbers and creates another number.
- **arithmetic shift operation** – A shift operation in which the sign of the number is preserved.
- **Boolean algebra** – An algebra for manipulation of objects that can take one of only two values: true or false.
- **circular shift operation** – A shift operation in which dropped bits from one end of a binary word are inserted at the other end.
- **logic operation** – An operation in which the result is a logical value: true or false.
- **logical shift operation** – A shift operation that does not preserve the sign of a number.
- **mask** – A variable or constant that contains a bit configuration used to control the setting of bits in a bitwise operation.
- **NOT operation** – An operation that complements bits: 0 becomes 1, or 1 becomes 0.
- **OR operation** – A binary operation that results in an output of 0 if both inputs are 0, otherwise 1.
- **rotate operation** – See *circular shift operation*.
- **set** – Force to 1, an application of the OR operator.
- **truth table** – A table listing all the possible logical input combinations, with all the corresponding logical output.
- **unset** – Force to 0, an application of the AND operator.
- **XOR** operation – A bitwise operation in which the result of the operation is 1 if one, and only one, of the operands is 1.

Summary

Operations on data can be divided into three broad categories: logic operations, shift operations, and arithmetic operations. Logic operations refer to those operations that apply the same basic operations to individual bits of a pattern or to two corresponding bits in two patterns. Shift operations move the bits in the pattern. Arithmetic operations involve adding, subtracting, multiplying, and dividing.

The four logic operators discussed in this chapter (NOT, AND, OR and XOR) can be used at the bit level or the pattern level. The NOT operator is a unary operator, while the AND, OR, and XOR operators are binary operators.

The only application of the NOT operator is to complement the whole pattern. One of the applications of the AND operator is to unset (force to 0) specific bits in a pattern. One of the applications of the OR operator is to set (force to 1) specific bits in a bit pattern. One of the applications of the XOR operator is to flip (complement) specific bits in a pattern.

Shift operations move the bits in a pattern: they change the positions of the bits. We can divide shift operations into two categories: logical shift operations and arithmetic shift operations. A logical shift operation is applied to a pattern that does not represent a signed number. Arithmetic shift operations assume that the bit pattern is a signed integer in two's complement format.

All arithmetic operations such as addition, subtraction, multiplication, and division can be applied to integers. Integers are normally stored in two's complement format. One of the advantages of two's complement representation is that there is no difference between addition and subtraction. When the subtraction operation is encountered, the computer simply changes it to an addition operation, but forms the two's complement of the second number. Addition and subtraction in sign-and-magnitude representation looks very complex. We have eight situations to consider.

All arithmetic operations such as addition, subtraction, multiplication, and division can be applied to reals stored in floating-point format. Addition and subtraction of real numbers stored in floating-point numbers is reduced to addition and subtraction of two integers stored in sign-and-magnitude after the alignment of decimal points.

--ooOoo--

Chapter 5 - Computer Organization

Key terms

- **address bus** - The part of the system bus used for address transfer.
- **address space** – A range of addresses.
- **arithmetic logic unit (ALU)** – The part of the computer system that performs arithmetic and logic operations on data.
- **bus** – The physical channel that links hardware components in a computer: the shared physical medium used in a bus-topology network.
- **cache memory** – A small, fast memory used to hold data items that are being processed.
- **central processing unit (CPU)** – The part of a computer that contains the control components to interpret instructions. In a personal computer, a microchip containing a control unit and an arithmetic logic unit.
- **compact disk read-only memory (CD-ROM)** – A compact disk in which data is written to the disk by the manufacturer and can only be read, and not written to, by the user.
- **compact disk rewritable (CD-RW)** – A compact disk that can be written to many times and read from many times.
- **complex instruction set computer (CISC)** – A computer that defines an extensive set of instructions, even those that are used less frequently.
- **control bus** – The bus that carries information between computer components.
- **control unit** – The component of a CPU that interprets the instructions and controls the flow of data.
- **data bus** – The bus inside a computer used to carry data between components.
- **decode** – The second phase in a machine cycle – fetch, decode, execute.
- **digital versatile disk (DVD)** – A direct access optical storage medium.
- **direct memory access (DMA)** – A form of I/O in which a special device controls the exchange of data between memory and I/O devices.
- **dynamic RAM (DRAM)** – RAM in which the cells use capacitors. DRAM must be refreshed periodically to retain its data. Slow but inexpensive.
- **electronically erasable programmable read-only memory (EEPROM)** – Programmable read-only memory that can be programmed and erased using electronic impulses without being removed from the computer.
- **erasable programmable read-only memory (EPROM)** – Programmable read-only memory that can be programmed. Erasing EPROM requires removing it from the computer.
- **execute** – The part of the instruction cycle in which the instruction which has been fetched and decoded is executed.
- **fetch** – The part of the instruction cycle in which the instruction to be executed is brought in from memory.

- **FireWire** – An I/O device controller with a high-speed serial interface that transfers data in packets.
- **input / output controller** – A device that controls access to input / output devices.
- **input / output subsystem** – The part of the computers' organization that receives data from the outside and sends data to the outside.
- **instruction register** – A register in the CPU that holds the instruction before being interpreted by the control unit.
- **interrupt-driven I/O** – A form of I/O in which the CPU, after issuing an I/O command, continues serving other processes until it receives an interrupt signal that the I/O operation is completed.
- **intersector gap** – The gap between sectors on a disk.
- **intertrack gap** – The gap between tracks on a tape.
- **isolated I/O** – A method of addressing an /O module in which the instructions used to read / write memory are totally different from the instructions used to read / write to input / output devices.
- **keyboard** – Non-storage input device.
- **land** - On an optical disk, an area not hit by the laser in the translation of a bit pattern. Usually represents a bit.
- **machine cycle** – Repeated phases, fetch, decode, execute, to execute instructions in the program.
- **magnetic disk** – A storage medium with random access capability.
- **magnetic tape** – A storage medium with sequential access capability.
- **main memory** – The primary memory of a computer, consisting of medium speed random access memory. Contrast with *cache memory*.
- **master disk** – First step in creating a CD-ROM.
- **memory mapped I/O** – A method of addressing an I/O module in a single address space, used for both memory and I/O devices.
- **monitor** – Non-storage input / output device.
- **multiple instruction-stream, multiple data-stream (MIMD)** – A computer architecture in which several instructions belonging to several instruction streams simultaneously operate on several data streams. (Each instruction on one data stream.)
- **multiple instruction-stream, single data-stream (MISD)** – A computer architecture in which several instructions belonging to several instruction streams simultaneously operate on the same data stream.
- **non-storage device** – allow the CPU / memory to communicate with the outside world, but they cannot store information.
- **optical storage device** – An I/O device that uses (laser) light to store and retrieve data.

- **parallel processing** – n/a
- **pit** – On an optical disk, an area struck by the laser in the translation of a bit pattern, which usually represents a 0 bit.
- **polycarbonate resin** – In CD-ROM production, a material injected into a mould.
- **printer** – A non-storage output device that creates a permanent record.
- **program counter** – A register in the CPU that holds the address of the next instruction in memory to be executed.
- **programmable read-only memory (PROM)** – Memory with contents electrically set by the manufacturer that may be reset by the user.
- **programmed I/O** – A form of I/O in which the CPU must wait for the I/O operation to be completed.
- **random access memory (RAM)** – The main memory of the computer that stores data and programs.
- **read-only memory (ROM)** – Permanent memory with contents that cannot be changed.
- **read / write head** – The device in a hard disk that reads or writes data.
- **reduced instruction set computer (RISC)** – A computer that uses only frequently used instructions.
- **register** – A fast stand-alone storage location that holds data temporarily.
- **rotational speed** – The spin rate of a magnetic disk.
- **sector** – A part of the track on a disk.
- **seek time** – In disk access, the time required to move the read / write head over the track that holds the required data.
- **single instruction-stream, multiple data-stream (SIMD)** – A computer architecture that has one control unit, multiple processing units, and one memory unit.
- **small computer system interface (SCSI)** – An I/O device controller with a parallel interface.
- **static ram (SRAM)** – A technology that uses traditional flip-flop gates (a gate with 2 states, 0 and 1) to hold data. Fast but expensive.
- **storage device** – An I/O device that can store large amounts of information for retrieval at a later time.
- **throughput** – n/a
- **track** – A part of a disk.
- **transfer time** – The time taken to move data from the disk to the CPU / memory.
- **Universal Serial Bus (USB)** – A serial I/O device controller that connects slower devices such as the keyboard and mouse to a computer.
- **write once, read many (WORM)** – Another name for a CD-R.

Summary

The parts that make up a computer can be divided into three broad categories or subsystems: the central processing unit (CPU), the main memory, and the input / output subsystem.

The central processing unit (CPU) performs operations on data. It has three parts: an arithmetic logic unit (ALU), a control unit, and a set of registers. The arithmetic logic unit (ALU) performs logic, shift, and arithmetic operations on data. Registers are fast stand-alone storage locations that hold data temporarily. The control unit controls the operation of each part of the CPU.

Main memory is a collection of storage locations, each with a unique identifier called the *address*. Data is transferred to and from memory in groups of bits called *words*. The total number of uniquely identifiable locations in memory is called the *address space*. Two types of memory are available: random access memory (RAM) and read-only memory (ROM).

The collection of devices referred to as the input / output (I/O) subsystem allows a computer to communicate with the outside world and to store programs and data even when the power is off. Input / output devices can be divided into two broad categories: non storage and storage devices. Non storage devices allow the CPU / memory to communicate with the outside world. Storage devices can store large amounts of information to be retrieved at a later time. Storage devices are categorized as either magnetic or optical.

The interconnection of the three subsystems of a computer plays an important role, because information needs to be exchanged between these subsystems. The CPU and memory are normally connected by three groups of connections, each called a *bus*: data bus, address bus, and control bus. Input / output devices are attached to the buses through an *input / output controller* or interface. Several kinds of controllers are in use. The most common ones today are SCSI, FireWire, and USB.

There are two methods of handling the addressing of /O devices: isolated I/O and memory-mapped I/O. In the isolated I/O method, the instructions used to read / write to and from memory are different from the instructions used to read / write to and from input / output devices. In the memory-mapped I/O method, the CPU treats each register in the I/O controller as a word in memory.

Today, general-purpose computers use a set of instructions called a *program* to process data. A computer executes the program to create output data from the input. Both the program and the data are stored in memory. The CPU uses repeating machine cycles to execute instructions in the program, one by one, from beginning to end. A simplified cycle can consist of three phases: *fetch*, *decode*, and *execute*.

Three methods have been devised for synchronization between I/O devices and the CPU: programmed I/O, interrupt-driven I/O, and direct memory access (DMA).

The architecture and organization of computers have gone through many changes during recent decades. We can divide computer architecture into two broad categories: CISC (complex instruction set computers), and RISC (reduced instruction set computers).

Modern computers use a technique called *pipelining* to improve their throughput. The idea is to allow the control unit to perform two or three phases simultaneously, which means that processing of the next instruction can start before the previous one is finished.

Traditionally, a computer had a single control unit, a single arithmetic logic unit, and a single memory unit. Parallel processing can improve throughput by using multiple instruction streams to handle multiple data streams.

--ooOoo--

Chapter 6 - Computer Networks

Key Terms

- **active document** – On the World Wide Web, a document at the local site using Java, or other scripting language.
- **Active Server Pages** (ASP) – A Microsoft technology that uses C# or VBasic to create active documents.
- **application layer** – The 7th layer in the TCP/IP model: provides access to network services.
- **backbone** – The bus that provides a link to all the devices in a bus topology network.
- **best-effort service** – IP protocol does not guarantee that all packets will arrive error-free, or in the order intended by the sender, or even be delivered.
- **bus topology** – A network topology in which all computers are attached to a shared medium.
- **chatting** – An Internet application.
- **client-server architecture** – The model of interaction between two application programs in which a program at one end (client) requests a service from a program at the other end (server).
- **cold fusion** – A dynamic web technology that allows the fusion of data items coming from a conventional database.
- **Common Gateway Interface** (CGI) – A standard for communication between HTTP servers and executable programs used in creating dynamic documents.
- **congestion control –** Any method to control excessive network or internetwork traffic causing a general degradation of service.
- **connection-oriented protocol** – A protocol for data transfer that establishes connection before transferring data.
- **connectionless protocol** – A protocol for data transfer without connection establishment or termination.
- **data link layer** – The second layer in the TCP/IP data model responsible for node-to-node delivery.
- **data-link layer address –** The address used in the data link layer, sometimes called the MAC address, sometimes the physical address.
- **demultiplexing** – Separating multiplexed data.
- **domain name** – In DNS, a sequence of labels separated by dots.
- **Domain Name Server** – A computer that holds information about Internet domain names.
- **dotted decimal notation** – The notation devised to make IP addresses easier to read: each byte is converted to a decimal number, numbers are separated by a dot.
- **dynamic document** – A web document created by running a program at a server site.
- **electronic mail (e-mail)** – A method of sending messages electronically based on a mailbox address rather than host-to-host exchange.

- **encapsulation** – The software engineering design concept in which data and their operations are bundled together and maintained separately from the application using them.
- **Extensible Markup Language (XML)** – A language that allows a user to describe the contents of a document. Also used as a query language in an object-oriented language.
- **frame** – A data unit at the data-link layer.
- **header** – The information added to the beginning of a packet for routing and other purposes.
- **hub** – A device that connects other devices in a network.
- **Hypertext Markup Language** – The computer language for specifying the contents and format of a web document: allows text to include fonts, layouts, embedded graphics, and links to other documents.
- **Hypertext Preprocessor** (PHP) – A scripting language.
- **Hypertext Transfer Protocol (HTTP)** – The protocol that is used to retrieve web pages on the Internet.
- **internet** – Abbreviation for internetwork.
- **Internet** – The global internet that uses the TCP/IP protocol suite.
- **Internet Mail Access Protocol (IMAP)** – A popular mail access protocol. Has more features and is more powerful than POP.
- **Internet Protocol (IP)** – The network layer protocol in the TCP/IP protocol responsible for transmitting packets from one computer to another across the internet.
- **internet service provider (ISP)** – An organization that provides Internet services.
- **internetwork** – A network of networks.
- **IP address –** A 32 bit address used to define a computer uniquely on the Internet (Internet address).
- **Java Applet** – A Java program saved on the server, compiled and ready to run.
- **Java Server Pages** – A technology used to create dynamic documents using scripts.
- **JavaScript** – scripting language used for active documents.
- **listserv** – An application program in the Internet.
- **local area network (LAN)** – A network connecting devices in a inside a limited area.
- **medium access control address (MAC) –** See *data link layer address*.
- **mesh topology** – A topology in which each device is connected to every other device.
- **message transfer agent** – An SMTP program that transfers a message across the internet.
- **metropolitan area network (MAN)** – A network that can span a city or a town.
- **multidrop connection –** See *multipoint connection*.

- **multiplexing** – The process of combining signals or data from multiple sources for transmission.
- **multipoint connection** – A connection in which 2 or more devices share the capacity of a link.
- **Multipurpose Internet Mail Extension (MIME)** – A supplement to SMTP that allows non-ASCII data to be sent through SMTP.
- **network** – A system of connected nodes that can share resources.
- **network layer** – The 3rd layer in the TCP/IP model, responsible for delivery of packets from the original host to the final destination.
- **node** – The devices in a network.
- **peer-to-peer communication** – An alternative to *client-server* communication.
- **performance** – Network criteria (Performance, reliability, security), measured by transit time and response time.
- **physical address –** The address of a device at the data-link layer.
- **physical layer** – The 1st layer in the TCP/IP model, responsible for signalling and transmitting bits across the network.
- **physical topology** – The way in which a network is laid out physically.
- **point-to-point connection** – A dedicated transmission link between two devices.
- **port number** – The address used in TCP and UDP to distinguish one process from another.
- **Post Office Protocol (POP)** – A popular mail access protocol that transfers e-mails from an e-mail server to the station of the e-mail recipient.
- **process –** A program in execution.
- **process-to-process communication** – Communication between 2 computers at the transport layer.
- **process-to-process delivery** – Delivery of a packet from the transport layer of the source computer to the transport layer of the destination computer.
- **reliability** – The quality factor that addresses the confidence or trust in a systems' total operation.
- **ring topology** – A topology in which the devices are connected in a ring. Each device receives a data unit from one neighbour and sends it to its other neighbour.
- **router** – A device operating at the first three TCP/IP layers that connects independent networks. A router routes a packet based on its destination address.
- **routing** – The process performed by a router.
- **security** – The quality factor that addresses the ease or difficulty with which an unauthorized user can access data.
- **Simple Mail Transfer Protocol** – The TCP/IP protocol for e-mail service.
- **star topology** – A topology in which all computers are connected to a common hub.

- **static document** – A web page that is created at the remote site and retrieved by the local site. Contrast with *dynamic document*.
- **Stream Control Transmission Protocol (SCTP)** – The transport layer protocol designed for Internet telephony and related applications.
- **TCP/IP protocol suite** – A five-layer protocol suite that defines the exchange of transmission across the internet.
- **terminal network** (TELNET) – A general-purpose client-server program that allows remote login.
- **trailer** – Control information appended to a data unit.
- **Transmission Control Protocol (TCP)** – One of the transport layer protocols in the TCP/IP protocol suite.
- **transport layer** – The 4th layer in the TCP/IP model, responsible for end-to-end delivery of the whole message.
- **Uniform Resource Locator (URL)** – A string of characters that defines a page on the internet.
- **user agent** (UA) – An SMTP component that prepares the message, creates the envelope, and puts the message in the envelope.
- **User Datagram Protocol (UDP)** – One of the transport layer protocols in the TCP/IP protocol suite.
- **video conferencing** – An application program on the Internet.
- **wide area network (WAN)** – A network that spans a large geographical distance.
- **World Wide Web (www)** – A multimedia Internet service that allows users to traverse the Internet by moving from one document to another via links.

Summary

A network is a combination of hardware and software that sends data from one location to another. We discussed two types of connection in a network: point-to-point and multipoint. We discussed four types of physical topologies in a network: mesh, star, bus, and ring.

We mentioned three types of networks: LAN, MAN, and WAN. A local area network (LAN) is usually privately owned and links the devices in a single office, building, or campus. A wide area network (WAN) provides long-distance transmission of data, images, audio, and video information over large geographic areas that may compromise a country, a continent, or even the whole world. A metropolitan area network (MAN) is a network with a size between a LAN and a WAN. When two or more networks are connected, they become an internetwork, or an internet.

The most notable internet is called the Internet, a collaboration of hundreds of thousands of interconnected networks. Most end users who want Internet connection use the services of Internet service providers

(ISP's). The set, or suite, of protocols that controls the Internet today is referred to as the TCP/IP protocol suite. The suite is made up of five layers.

The application layer enables the user to access the network. It provides support for services such as electronic mail, remote file access and transfer, browsing the World Wide Web, and so on. The addresses on the application layer are specific to the application programs.

The transport layer is responsible for process-to-process delivery of the entire message; this means that a logical communication is created between the transport layers of the client and the server computer. During the life of the TCP/IP protocol suite, three transport layer protocols have been designed: UDP, TCP, and SCTP.

The network layer is responsible for the source-to-destination delivery of packet, possibly across multiple networks. The network layer ensures that each packet gets from its point of origin to its final destination. In the TCP/IP protocol suite, the main protocol at the network layer is Internet Protocol (IP).

The data-link layer delivers a packet from one node to another. The data-link layer is also responsible for error and flow control between “hops”. It uses physical or MAC address to identify the nodes.

The physical layer coordinates the functions required to carry a bit stream over a physical medium. Although the data-link layer is responsible for moving a frame from one node to another, the physical layer is responsible for moving the individual bits that enable the frame to reach the next node.

Electronic mail (e-mail) is the most popular application on the Internet. The main protocol used for e-mail is called Simple Mail Transfer protocol (SMTP). Other protocols used for electronic mail are POP and IMAP.

File Transfer Protocol (FTP) is the standard mechanism for one of the most common tasks on the internet, copying a file from one computer to another. FTP differs from other client-server applications in that it establishes two connections: one for data transfer and one for exchanging control commands.

TELNET is a general-purpose client-server program that lets a user access any application program on a remote computer. In other words, it allows the user to log in to a remote computer. After login, a user can use the services available on the remote computer and transfer the results back to the local computer.

The World Wide Web (WWW), or web, is a repository of information spread all over the world and linked together. To use the WWW, we need three components: a browser, a web server, and a protocol called Hypertext Transfer Protocol (HTTP).

Documents on the WWW can be grouped into three broad categories: static, dynamic, and active. The

category is based on the time at which the contents of the document are determined. A dynamic document is created by a web server whenever a browser requests the document. An active document is a program that is run at the client site.

Three other applications were mentioned in this chapter. Video conferencing can provide communication between two or more groups of participants or a set of individual participants. Listservs allow a group of users to discuss a common topic of interest by using two programs running on the server: the subscriber server and the mailer server. Finally, chat is supported by a class of real-time applications similarly to video conferencing, in which two or more parties are involved in an exchange of text, audio, and video.

--ooOoo--

Chapter 7 - Operating Systems

Key Terms

- **batch operating system** – The operating system used in early computers, in which jobs were grouped before being served.
- **bootstrap** – The process in which the operating system is loaded into main memory when the computer is turned on.
- **circular waiting** – A condition in an operating system in which all processes and resources involved form a loop.
- **deadlock** – A situation in which the resources needed by one job to finish its task are held by another job.
- **demand paging** – A memory allocation method in which a page of a program is loaded into memory only when it is needed.
- **demand paging and segmentation** – A memory allocation method in which a page or a segment of a program is loaded into memory only when it is needed.
- **demand segmentation** – A memory allocation method in which a segment of a program is loaded into memory only when it is needed.
- **device manager** – A component of an operating system that controls access to the input / output devices.
- **distributed system** – An operating system that controls resources located in computers at different sites.
- **frame** – In *paging*, memory is divided into equally sized sections called frames.
- **graphical user interface** (GUI) – A user interface that defines icons and operations on icons.
- **hold state** – The state of a job that is waiting to be loaded into memory.
- **job** – A program becomes a job when it is selected for execution. Can be in *hold* or *terminated* state.
- **job scheduler** – A scheduler that selects a *job* for processing from a queue of jobs waiting to be moved into memory.
- **kernel –** The main part of an operating system.
- **Linux** – An operating system developed by Linus Torvalds to make UNIX more efficient when run on an Intel microprocessor.
- **memory management** – The component of the operating system that controls the use of main memory.
- **Microsoft Disk Operating System** (MS-DOS) – The operating system based on DOS, developed by Microsoft.
- **mono programming** – The technique that allows only one program to be in memory at a time.
- **multi-programming** – A technique that allows more than one program to reside in memory while being processed.

- **mutual exclusion** – A condition imposed by an operating system in which only one process can hold a resource.
- **native NT file system** (NTFS) – The standard file system of Windows NT, and later versions.
- **no pre-emption** – A condition in which the operating system cannot temporarily allocate a resource.
- **operating system** – An interface between the hardware of a computer and the user that facilitates the execution of other programs and the access to hardware and software resources.
- **page** – One of a number of equally sized sections of a program.
- **paging** - A multi-programming technique in in which memory is divided into equally sized sections called *frames*.
- **parallel system** – An operating system with multiple CPU's on the same machine.
- **partitioning** – A technique used in multi-programming that divides the memory into variable-length sections.
- **process** – A program in execution. Can be in ready, running or waiting states.
- **process scheduler** – An operating system mechanism that despatches the processes waiting to get access to the CPU.
- **program** - A set of instructions.
- **queuing** – The system whereby jobs or processes waiting for resources are handled.
- **ready state** – In process management, the state of processing in which the process is waiting to get the attention of the CPU.
- **real-time system** – An operating system that is expected to do a task within specific time constraints.
- **resource holding** – A condition in which a process holds a resource but cannot use it until all other required resources are available.
- **running state** – In process management, a state in which a process is using the CPU.
- **scheduler** – A program to move a job from one state to another.
- **scheduling** – Allocating the resources of an operating system to different programs and deciding which program should use which resource, and when.
- **shell** – A user interface in some operating systems, such as UNIX.
- **single-user operating system** – An operating system in which only one program can be in memory at a time.
- **software** – The application and system programs necessary for computer hardware to accomplish a task.
- **starvation** – A problem in the operation of an operating system in which processes cannot get access to the resources they need.

- **state diagram** – A diagram that shows the different states of a process.
- **terminal state** – The last state in a state diagram.
- **time sharing** – An operating system concept in which more than one user has access to a computer at the same time.
- **UNIX** – A popular operating system among computer programmers and computer scientists.
- **user interface** – A program that accepts requests from users (processes) and interprets them for the rest of the operating system.
- **utility** – A type of application program in UNIX.
- **virtual memory -** A form of memory organization that allows swapping of programs between memory and magnetic storage to give the impression of a larger main memory than really exists.
- **waiting state** – A state in which a process is waiting to receive the attention of the CPU.
- **window** – A user interface (Shell in UNIX).
- **Windows 2000** – A version of the Windows NT operating system.
- **Windows NT** – An operating system devised by Microsoft to replace MS-DOS.
- **Windows XP** – A version of the Windows NT operating system.

Summary

An operating system is an interface between the hardware of a computer and the user, that facilitates the execution of programs and access to hardware and software resources. Two major design goals of an operating system are efficient use of hardware and ease of use of resources.

Operating systems have gone through a long history of evolution; batch systems, time-sharing systems, personal systems, parallel systems, and distributed systems.

A modern operating system has at least four functional areas: memory manager, process manager, device manager, and file manager. An operating system also provides a user interface.

The first responsibility of a modern computer system is memory management. Memory allocation must be controlled by the operating system. Memory management techniques can be divided into two categories: mono programming and multiprogramming. In mono programming, most of the memory capacity is dedicated to one single program. In multiprogramming, more than one program can be in memory at the same time.

The second responsibility of an operating system is process management. A process is a program in execution. The process manager uses schedulers and queues to manage processes. Process management

involves synchronizing different processes with different resources. This may potentially create resource deadlock or starvation. Deadlock occurs when the operating system does not put resource restrictions on processes; starvation can happen when the operating system puts too many resource restrictions on a process.

The third responsibility of an operating system is device or input / output management. There are limitations on the number and speed of input / output devices in a computer system. Because these devices are much slower compared to the CPU and memory, when a process accesses an input / output device, it is not available to other processes. The device manager is responsible for the efficient use of input / output devices.

The fourth responsibility of an operating system is file management. An operating system uses a file manager to control access to files. Access is permitted only by processes or users that are allowed access to specific files, and the type of access can vary.

Two common operating systems with some similarities are UNIX and Linux. UNIX is a multi-user, multiprocessing, portable operating system made up of four parts: the kernel, the shell, a standard set of utilities, and application programs. Linux has three components: a kernel, system utilities, and a system library.

A popular family of operating systems from Microsoft is referred to as Windows NT. Windows NT is an object-oriented, multi-layer operating system. It uses several layers, including a hardware abstract layer (HAL), executive layer, and an environment subsystem layer.

--ooOoo--

Chapter 8 - Algorithms

Key Terms

- **algorithm** – An ordered set of unambiguous steps that produce a result and terminates in a finite time.
- **binary search** – A search algorithm in which the search value is located by repeatedly dividing the list in half.
- **bubble sort** – A sort algorithm in which each pass through the data moves (bubbles) the lowest element to the beginning of the unsorted portion of the list.
- **decision** – One of 3 structured program, or algorithm, constructs. (Sequence, decision, repetition).
- **input data** – User information that is submitted to a computer to run a program.
- **insertion sort** – A sort algorithm in which the first element from the unsorted portion of the list is inserted into its proper position in the sorted portion of the list.
- **loop** – A structured programming construct that causes one or more statements to be repeated.
- **output data** – The results of running a computer program.
- **product** – One of the basic algorithms. (Summation, product, smallest / largest, sorting, searching).
- **pseudocode** – English-like statements that follow a loosely defined syntax and are used to convey the design of an algorithm or function.
- **recursion** – A function design in which the function calls itself.
- **repetition** – One of the 3 constructs in structured programming.
- **searching** - Basic algorithm - the process that examines a list to locate one or more elements containing a designated value known as the search argument.
- **selection** – See *decision*.
- **selection sort** – A sort algorithm in which the smallest value in the unsorted portion of the list is selected and placed at the end of the sorted portion of the list. (Swapped with the first element of the unsorted portion of the list).
- **sequence** – One of the 3 constructs in structured programming.
- **sequential search** – A search technique used with a linear list in which searching begins at the first element and continues until the value of an element equal to the value being sought is located, or until the end of the list is reached.
- **sort pass** – One loop during which all elements are tested by a sorting program.
- **sorting** - The process that orders a list or file.
- **structure chart** – A design and documentation tool that represents a program as a hierachial flow of functions.
- **sub algorithm** – A part of an algorithm that is independently written and is executed when called inside the algorithm.

- **summation** – One of the basic algorithms.
- **Unified Modelling Language (UML)** – A graphical language used for analysis and design.

Summary

An algorithm can be informally defined as “a step-by-step method for solving a problem or doing a task”. More formally, an algorithm is defined as “an ordered set of unambiguous steps that produce a result and terminates in a finite time”.

Computer scientists have defined three constructs for a structured program or algorithm: *sequence*, *decision* (selection), and *repetition* (loop).

Several tools have been designed to show an algorithm: UML, pseudocode, and structure charts. UML is a pictorial representation of an algorithm. Pseudocode is an English-language-like representation of an algorithm. A structure chart is a high-level design tool that shows the relationship between algorithms and sub-algorithms.

Several algorithms are used in computer science so prevalently that they are considered basic. We discussed the most common in this chapter: *summation*, *product*, *finding the smallest and largest*, *sorting*, and *searching*.

One of the most common applications in computer science is sorting, which is the process by which data is arranged according to its value. we introduced three primitive, but fundamental, sorting algorithms: *selection sort*, *bubble sort*, and *insertion sort*. These three sorting algorithms are the foundation of the faster sorts used in computer science today.

Another common algorithm in computer science is searching, which is the process of finding the location of a target among a list of objects. There are two basic searches for lists: *sequential search*, and *binary search*. Sequential search can be used to locate an item in any list, whereas binary search requires the list to be sorted.

The principles of structured programming require that an algorithm be broken into small units called sub-algorithms. Each sub-algorithm is in turn divided into smaller sub-algorithms

In general, there are two approaches to writing algorithms to solve a problem. One uses *iteration*, the other uses *recursion*. An algorithm is iterative whenever the definition does not involve the algorithm itself. An algorithm is defined recursively whenever the algorithm appears within the definition itself.

--ooOoo--

Chapter 9 - Programming Languages

Key Terms

- **actual parameter** – The parameters in the function calling statement that contain the values to be passed to the function. Contrast with *formal parameters*.
- **Ada** – A high-level concurrent programming language developed by the US Department of Defence.
- **applet** – A computer program written in Java (or other scripting language) that creates an active web document.
- **arithmetic operator** – The operator used in arithmetic operations.
- **assembler** – System software that converts a source program into executable object code. Traditionally associated with an assembly language program. See *compiler*.
- **assembly language** – A programming language in which there is a one-to-one correspondence between the computers' machine language and the symbolic instruction set of the language.
- **assignment statement** – The statement that assigns a value to a variable.
- **bytecode** – A machine language into which a Java source program is compiled.
- **C language** – A procedural language developed by Dennis Ritchie.
- **C++ language** – An object-oriented language developed by Bjarne Stroustrup.
- **class** – The combination of data and functions to form a type.
- **code generator** – A process in a compiler or interpreter that creates the machine language code.
- **COmmon Business-Oriented Language (COBOL)** – A business programming language developed by Grace Hopper.
- **compilation** – The process of translating the whole source program written in a high-level language into machine language before executing the program.
- **compiler** – System software that converts a source program into executable object code: traditionally associated with high-level languages. See also *assembler*.
- **composite type** – A data type that is composed of two or more simple types.
- **compound statement** – In some programming languages, a collection of statements (instructions) treated as one by the language.
- **computer language** – Any of the syntactical languages used to write programs for computers, such as machine language, assembly language, C, COBOL, and FORTRAN.
- **constant** – A data value that cannot change during the execution of the program. Contrast with *variable*.
- **control statement** – A statement that alters the sequential flow of control in a source program.
- **data type** – A named set of values and operations defined to manipulate them, such as character and integer.

- **declarative paradigm** – A paradigm that uses the principle of logical reasoning to answer queries.
- **expression** – A sequence of operators and operands that reduces to a single value.
- **formal parameter** – The parameter declaration in a function to describe the type of data to be passed to the function.
- **FORmula TRANslation (FORTRAN)** – A high-level procedural language used for scientific and engineering applications.
- **functional language** – A programming language in which a program is considered to be a mathematical function.
- **functional paradigm** – A paradigm in which a program is considered to be a mathematical function.
- **high-level language** – A (portable) programming language designed to allow the programmer to concentrate on the application rather than the structure of a particular computer or operating system.
- **identifier** – The name given to an object in a programming language.
- **imperative paradigm** (opposite of declarative) – another name for procedural paradigm.
- **inheritance** – The ability to extend a class to create a new class while retaining the data objects and methods of the base class, and adding new data objects and methods.
- **input** – n/a
- **interpretation** – The process of translating each line of the source program into the corresponding line of the object program and executing the line.
- **interpreter** – A program that translates a source program in a high-level language line by line and executes each line immediately.
- **Java** - An OOP programming language.
- **lexical analyzer** – A program used in a translation process that reads the source code symbol by symbol and creates a list of tokens.
- **LISt Programming Language (LISP)** – A list processing programming language in which everything is considered a list. A *functional* language.
- **literal** – A constant used in a program.
- **local variable** – A variable defined within a block or a module.
- **logical operator** – n/a
- **machine language** – Instructions native to the central processor of a computer that are executable without assembly or compilation.
- **method** – A (virtual) function in an object-oriented language.
- **multi-threading** – Parallel processing supported by some languages such as Java.
- **object program** – The machine language code created from a compiler or interpreter.

- **object-oriented language** – A programming language in which the objects and the operations to be applied to them are tied together. (Encapsulation)
- **object-oriented paradigm** – A paradigm in which a program acts on active objects.
- **operand** – An object in a statement on which an operation is performed.
- **operator** – The syntactical token representing an action on data (the operand).
- **output** – The results of running a computer program.
- **Pascal** – A programming language designed with the goal of teaching programming to novices by emphasizing the structured programming approach.
- **pass by reference** – A parameter passing technique in which the called function refers to a passed parameter using an alias name.
- **pass by value** – A parameter passing technique in which the the value of a variable is passed to a function.
- **polymorphism** – In C++, defining several operations with the same name that can do different things in related classes.
- **procedural paradigm** – A paradigm in which a program acts on passive objects using procedures.
- **Programming in LOGic (PROLOG)** – A declarative programming language that can build a database of facts and a knowledge base of rules.
- **relational operator** – n/a
- **semantic analyzer** – Analyses the meaning of words in a sentence or tokens in a statement.
- **simple type** – A data type such as integer or real.
- **source program** – The file that contains program statements written by a programmer before they are converted into machine language: the input file to an assembler or compiler.
- **statement** – A syntactical construct in C that represents one operation in a function.
- **sub-program** – A smaller program called by the main program.
- **syntax** – The grammatical rules of a language.
- **syntax analyzer** – The process that checks the grammar of a sentence.
- **token** – A syntactical construct that represents an operation, a flag, or a piece of data.
- **variable** – A memory storage object whose value can be changed during the execution of a program.

.

Summary

A computer language is a set of predefined words that are combined into a program according to predefined rules, the languages' syntax. Over the years, computer languages have evolved from machine language to high-level languages. The only language understood by a computer is machine language.

High-level languages are portable to many different computers, allowing the programmer to concentrate on the application rather than the intricacies of the computers' organization.

To run a program on a computer, the program needs to be translated into the computers' native machine language. The program in the high-level language is called the *source program*. The translated program in machine language is called the *object program*. Two methods are used for translation: *compilation* and *interpretation*. A compiler translates the whole source program into the object program. Interpretation refers to the process of translating each line of the source program into the corresponding object program line by line and executing them.

The translation process uses a lexical analyzer, a syntax analyzer, a semantic analyzer, and a code generator to create a list of tokens.

A paradigm describes a way in which a computer language can be used to approach a problem to be solved. We divide computer languages into four paradigms: *procedural*, *object-oriented*, *functional*, and *declarative*.

The procedural paradigm considers a program as an active agent that manipulates passive objects. FORTRAN, COBOL, Pascal, C, and Ada are examples of procedural languages.

The object-oriented paradigm deals with active objects instead of passive objects. C++ and Java are common object-oriented languages.

In the functional paradigm, a program is considered as a mathematical function. In this context, a function is a black-box that maps a list of inputs to a list of outputs. LISP and Scheme are common functional languages.

A declarative paradigm uses the principle of logical reasoning to answer queries. One of the best-known declarative languages is PROLOG.

Some common concepts in procedural and object-oriented languages are: *identifiers*, *data types*, *variables*, *literals*, *constants*, *inputs*, and *outputs*, *expressions*, and *statements*. Most languages use two categories of control statements: *decision* and *repetition*. *Sub-programming* is a common concept among procedural languages.

--ooOoo--

Chapter 10 - Software Engineering

Key Terms

- **analysis phase** – A phase in the software system lifecycle that defines requirements that specify what the proposed system is to accomplish.
- **basis path testing** – A white-box test method that creates a set of test cases that executes every statement in the software at least once.
- **black-box testing** – Testing based on the system requirements rather than knowledge of the program.
- **class diagram** – A diagram that manifests the relationship between classes in a system.
- **cohesion** – The attribute of a module that describes how closely the processes in a module are related to one another.
- **data flow diagram** – A diagram that shows the movement of data in the system.
- **design phase** – A phase in the software system lifecycle that defines how the system will accomplish what was defined in the analysis phase.
- **development process** – The process of developing software that is outside the system lifecycle. (Development, use, modification).
- **documentation** – Three types of documentation required for software to be used properly and maintained efficiently: User, system, and technical. Documentation only stops when the software becomes obsolete.
- **entity-relationship diagram** – A diagram used in entity-relationship modelling.
- **glass-box testing** – See *white-box testing*.
- **implementation phase** – A phase in the software system lifecycle in which the actual programs are created.
- **incremental model** – a model in software engineering in which the entire package is constructed with each module consisting of just a shell: modules gain complexity with each iteration of the package.
- **maintainability** – A quality that refers to keeping a system running correctly and up to date. (Changeability, Correctability, Flexibility, testability).
- **modularity** – Breaking a large project into small parts that can be understood and handled easily.
- **object-oriented analysis** – The analysis phase of a developmental process in which the implementation uses an object-oriented language.
- **object-oriented design** – The design phase of a developmental process in which the implementation uses an object-oriented language.
- **operability** – The quality factor that addresses the ease with which a system can be used. (Accuracy, efficiency, reliability, security, timeliness, usability).
- **procedure-oriented analysis** – The analysis phase of a developmental process in which the implementation uses a procedural language.

- **procedure-oriented design** – The design phase of a developmental process in which the implementation uses a procedural process.
- **software engineering** – The design and writing of structured programs.
- **software life cycle** – The life of a software package. (Development, use, modification).
- **software quality** – Can be divided into 3 broad measures: *operability*, *maintainability*, and *transferability*.
- **state chart** – A diagram, similar to a *state diagram*, but used in object-oriented software engineering.
- **state diagram** – A diagram that shows the different states of a process.
- **structure chart** – A design and documentation tool that represents a program as a hierachial flow of functions.
- **testing phase** – A phase n the software lifecycle in which experiments are carried out to prove that a software package works.
- **transferability** – A quality in software systems that refers to the ability to move the system from one platform to another. (Reusability, interoperability, portability).
- **use case diagram** – A diagram showing the user view of a system in UML.
- **waterfall model** – A software development model in which each module is completely finished before the next module is started.
- **white-box testing** – Program testing in which the internal design of the program is considered. Also known as glass-box testing. Contrast to *black-box testing*.

.

Summary

The software lifecycle is a fundamental concept in software engineering. Software, like many other products, goes through a cycle of repeating phases.

The development process in the software lifecycle involves four phases: analysis, design, implementation, and testing. Several models have been used in relation to these phases. we discussed the two most common: the waterfall model and the incremental model.

The development process starts with the analysis phase. The analyst prepares a specification document that shows what the software will do without specifying how it will be done. The analysis phase can be done in two ways: procedure-oriented analysis, and object-oriented analysis.

The design phase defines how the system will accomplish what was defined in the analysis phase. In procedure-oriented design, the whole project is divided into a set of procedures or modules. In object-oriented design, the design phase continues by elaborating the details of classes.

Modularity means breaking a large project into smaller parts that can be understood and handled easily. Two issues are important when a system is divided into modules: coupling and cohesion. Coupling is a measure of how tightly two modules are bound to each other. Coupling between modules in a software system must be minimized. Cohesion is a measure of how closely the modules in a system are related. Cohesion between modules in a software system should be maximized.

In the implementation phase, programmers write the code for the modules in procedure-oriented design, or write the program units to implement classes in the object-oriented design.

The quality of software is important. Software quality can be divided into three broad measures: operability, maintainability, and transferability.

The goal of the testing phase is to find errors. There are two types of testing: glass-box and black-box. Glass-box testing (or white-box testing) is based on knowing the internal structure of the software. Glass-box testing assumes that the tester knows everything. Black-box testing means testing the software without knowing what is inside it, and without knowing how it works.

--ooOoo--

Chapter 11 - Data Structures

Key Terms

- **array** – A fixed-size, sequenced collection of elements of the same data type.
- **column-major storage** – A method of storing two-dimensional arrays in which the elements are stored column.
- **data structure** – The syntactical representation of data organized to show the relationship among the individual elements.
- **field** – The smallest named unit of data that has meaning in describing information. A field may be either a variable or a constant.
- **index** – The address of an element in an array.
- **link** – The field that identifies the next element in the list, in a list structure.
- **linked list** – A linear list structure in which the ordering of the elements is determined by link fields.
- **multidimensional array –** An array with elements having more than one level of indexing.
- **node** – An element, in a data structure, that contains both data and structural elements used to process the data structure.
- **null pointer** – A pointer that points to nothing.
- **three-dimensional array** – An array with only one level of indexing.
- **pointer** – A constant or variable that contains an address that can be used to access data stored elsewhere.
- **record** – Information related to one entity.
- **row-major storage** – A method of storing array elements in memory in which the elements are stored row by row.
- **two-dimensional array** – An array with elements having two levels of indexing.

Summary

A data structure uses a collection of related variables that can be accessed individually, or as a whole. In other words, a data structure represents a set of data items that share a specific relationship. we discussed three data structures in this chapter: *arrays*, *linked lists*, and *records*.

An array is a sequenced collection of elements normally of the same data type. We use indexes to refer to the elements of an array. In an array we have two types of identifiers: the name of the array, and the name of each individual element.

Many applications require that data is stored in more than one dimension. One common example is a table,which is an array that consists of rows and columns. Two-dimensional arrays can be stored in memory using either row-major or column-major storage. The first is more common.

The common operations on arrays as a structure are: *searching*, *insertion*, *deletion*, *retrieval*, and *traversal*. An array is a suitable structure in applications where the number of deletions and insertions is small but a lot of searching and retrieval operations are required. An array is normally a static data structure and so is more suitable when the numbers of data items is fixed.

A record is a collection of related elements, possibly of different types, having a single name. Each element of a record is called a *field*. A field is the smallest element of named data that has meaning in a record.

A linked list is a collection of data in which each element contains the location of the next element; that is, each element contains two parts: *data* and *link*. The data part holds the useful information: the data to be processed. The link is used to chain the data together.

The same operations defined for an array can be applied to a linked list. A linked list is a very efficient structure for data that will go through many insertions and deletions. A linked list is a dynamic data structure in which the list can start with no nodes and grow as new nodes are needed.

--ooOoo--

Chapter 13 - File Structures

Key Terms

- **absolute pathname** – In Unix or Linux, a path name that starts from root.
- **access method** – A technique for reading data from a secondary (auxiliary) storage device.
- **auxiliary storage** – Any storage device outside main memory: permanent data storage, external storage, secondary storage.
- **binary file** – A collection of data stored in the internal format of the computer. (Compare to *text file*).
- **bucket** – In a hashing algorithm, a location that can accommodate multiple data units.
- **bucket hashing** – A hashing method that uses buckets to reduce collision.
- **collision** – In hashing, an event that occurs when a hashing algorithm produces an address for an insertion, and that address is already occupied.
- **collision resolution** – An algorithm process that determines an alternative address after a collision.
- **current directory** – In Unix and Linux, the directory that a user is in at the present time.
- **data file** – A file that contains only data, not programs.
- **digit extraction method** – A hashing method that used digit extraction.
- **direct hashing** – A hashing method in which the key is obtained without algorithmic modification.
- **directory** – A file that contains the names and addresses of other files.
- **division remainder method** – A type of hashing in which the key is divided by a number and the remainder is used as the address.
- **error report file** – In a file update process, a report of errors detected during the update.
- **hashed file** – A file that is searched using one of the hashing methods.
- **hashing method** – A method to access a hashed file.
- **home address** – In a hashed list, the address produced by the hashing algorithm.
- **home directory** – In Unix or Linux, a directory that a user is in when first logged in.
- **index** – A small file with 2 fields: the key of the sequential file, and the address of the corresponding record on the disk.
- **indexed file** – A file that uses an index for random access.
- **inverted file** – An indexed file with more than index, each with a different key.
- **key** – One or more fields used to identify a record (structure),
- **linked list resolution** – A collision resolution method in hashing that uses a separate area for synonyms, which are maintained in a linked list.

- **modulo division** – Dividing 2 numbers and keeping the remainder.
- **new master file** – The master file that is created from an old master file when the file is updated.
- **old master file** – The master file that is processed in conjunction with the *transaction file* to create the *new master file*.
- **open addressing resolution** – A collision resolution method whereby the data is placed in the first available address in the *prime area*.
- **parent directory** – the directory immediately above the *working directory*.
- **path** – A sequence of nodes in which each vertex is adjacent to the next.
- **prime area** – The part of the file that contains all the *home addresses*.
- **random access** – A storage method that allows data to be retrieved in an arbitrary manner.
- **relative pathname** – The path relative to the *working directory*.
- **root directory** – is the highest level in the file system hierarchy.
- **secondary storage** – See auxiliary *storage*.
- **sequential access** – An access method in which the records in a file are accessed serially, beginning with the first element.
- **sequential file** – A file structure in which data must be processed sequentially from the first element in the file.
- **synonym** – In a hashed list, two or more keys that hash to the same home address.
- **text file** – A file in which all data is stored as characters. Contrast with *binary file*.
- **transaction file** – A file containing relatively transient data that are used to change the contents of a master file.
- **working directory** – is the directory you are in at any point. See *current directory*.

Summary

A file is an external collection of related data treated as a single unit. The primary purpose of a file is to store data. Since the contents of main memory are lost when the computer is shut down, we need files to store data in a more permanent form. Files are stored in auxiliary or secondary storage devices.

The access method determines how records can be retrieved: sequentially or randomly. If we need to access a file sequentially, we use a sequential file structure. If we need to access one specific record without having to retrieve all records before it, we use a random file structure.

A sequential file is one in which records can only be accessed sequentially – one after another – from beginning to end. Sequential files must be updated periodically to reflect changes in information. There are four files associated with an update program: the new master file, the old master file, the transaction file, and the error report file.

To access a record in a file randomly, we need to know the address of the record. Two types of files are normally used for accessing records randomly: an indexed file, and a hashed file.

An indexed file is made up of a data file, which is a sequential file, and an index. The index itself is a very small file with only two fields: the key of the sequential file and the address of the corresponding record on disk. The index is sorted based on the key values of the data files. In a hashed file, the key is mapped to the record address using a hashing function.

Several methods have been used for hashing. In the direct method, the key is the address without any algorithmic manipulation. In the modulo division method, the key is divided by the file size and the remainder plus 1 is used for the address. In digit extraction hashing, selected digits are extracted from the key and used as the address.

In hashing there is a possibility that more than one key will hash to the same address in the file, resulting in a collision. We discussed a few collision resolution methods: open addressing, linked list resolution, and bucket hashing.

Directories are provided by most operating systems for organizing files. A directory performs the same function as a folder in a filing cabinet. However, a directory in most operating systems is represented as a special type of file that holds information about other files.

A file stored on a storage device is a sequence of bits that can be interpreted by an application program as a text file or a binary file. A text file is a file of characters. A binary file is a collection of data stored in the internal format of the computer.

--ooOoo--

Chapter 14 - Databases

Key Terms

- **attribute** - each column in a *relation*.
- **cardinality** - the number of *tuples* (rows) in a *relation* (table).
- **conceptual level** – Relating to the logical structure of a database. It deals with the meaning of the database, not its physical implementation.
- **database –** A collection of organized information.
- **database management system (DBMS)** – A program or a set of programs that manipulates a database.
- **database model** – A model that defines the logical design of data.
- **delete operation** – In a relational database, a unary operation that deletes a *tuple* from the *relation.* (SQL)
- **difference operation** – A binary operation on 2 sets, the result of which is the first set minus the common elements in the 2 sets, or an operator in a relational database that is applied to 2 relations with the same attributes. The tuples in the resulting relation are those that are in the first relation but not the second. (SQL)
- **distributed database –** A database in which data is stored on several computers.
- **Entity-Relation Diagram (ERD)** – A diagram used in entity-relationship modelling.
- **Entity-Relationship Model (ERM)** – A model that defines the entities and their relationship in a relational database.
- **external level** – The part of the database that interacts with the user.
- **fragmented distributed database** – A distributed database in which the data is localized.
- **hierarchical model** – A database model that organizes data in a treelike structure that can be searched from top to bottom. Obsolete.
- **insert operation –** A unary operation in a relational database that inserts a *tuple* into a *relation*. (SQL)
- **internal level** – The part of the database that defines where data is actually stored.
- **intersection operation –** A binary operation on 2 sets in which the result is a set with the elements common to the 2 sets.
- **join operation –** A binary operation in a relational database that takes 2 relations, which must have the same attributes, and combines them. Each tuple in the resulting relation is a member of both original relations. (SQL)
- **network model** – A database model in which a record can have more than one parent record. Obsolete.
- **Normal Form (NF)** – A step in the normalization process of a relational database.

- **normalization** – In a relational database, the process of applying normal forms to a relational model.
- **object-oriented database** – A database in which data is treated as structures (objects).
- **project operation –** A unary operation in a relational database in which a set of columns is selected based on a criterion. The attributes in the resulting relation are a subset of the attributes in the original relation. (SQL)
- **relation -** a two dimensional table. Has a *name, attributes* (columns)*, tuples* (rows).
- **relation database management system (RDBMS)** – A set of programs that handles relations in a relational database model.
- **relational model** – A database model in which data is organized in related tables called relations.
- **replicated distributed database** – A database in which each site holds a replica of another site.
- **select operation –** A unary operation in a relational database that selects a set of tuples, which are a subset of the original relation. (SQL)
- **Structured Query Language (SQL)** – A declarative database language that includes statements for database definition, manipulation, and control.
- **tuple -** each row in a *relation* (table) is called a tuple.
- **union operation –** A binary operation on 2 sets in which the result contains all the elements from both sets without duplicates. (SQL)
- **update operation** - A unary operation that changes the value of some attributes of a tuple. (SQL)

Summary

A database is a collection of data that is logically, but not necessarily physically, coherent – its various parts can be physically separated. A database management system (DBMS) defines, creates, and maintains a database.

The American National Standards Institute / Standards Planning and requirements Committee (ANSI / SPARC) has established a three-level architecture for a DBMS: internal, conceptual, and external. The internal level determines where data is actually stored on storage devices. The conceptual level defines the logical view of the data. The external level interacts directly with the user.

Traditionally, three types of database model were defined: hierarchical, network, and relational. Only the last, relational model, has survived.

In the relational model, data is organized in two dimensional tables called relations. A relation has the following features: name, attributes, and tuples.

In a relational database we can define several operations to create new relations based on existing ones. we mentioned nine operations in the context of the database query language SQL (Structured Query Language): insert, delete, update, select, project, join, union, intersection, and difference.

The design of a database, for example, for an organization, is often a lengthy task that can only be done through a step-by-step process. The first step often involves interviewing potential users of the database to collect the information that needs to be stored. The second step is to build an Entity-Relationship-Model (ERM) that defines the entities for which information must be maintained. The next step is to build relations based on the ERM.

Normalization is the process by which a given set of relations are transformed to a new set of relations with a more solid structure. Normalization is required to allow any relation in the database to be represented, to allow a query language such as SQL to use powerful retrieval operations composed of atomic operations, to remove anomalies in insertion, deletion, and updating, and to reduce the need for restructuring the database as new data types are to be added.

The relational database is not the only model of database in use today. The other two common models are distributed databases and object-oriented databases.

--ooOoo--

Appendix E - Boolean Algebra & Logic Gates

Boolean Algebra

Deals with variables and constants that can take only one of two values: 1 or 0.

Logic Gate – an electronic device that takes 1 to N inputs, and creates 1 output.

Buffer	
x	x
0	0
1	1

NOT	
x	x'
0	1
1	0

AND		
x	y	x.y
0	0	0
0	1	0
1	0	0
1	1	1

NAND		
x	y	(x.y)'
0	0	1
0	1	1
1	0	1
1	1	0

OR		
x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

NOR		
x	y	(x+y)'
0	0	1
0	1	0
1	0	0
1	1	0

XOR		
x	y	(x©y)
0	0	0
0	1	1
1	0	1
1	1	0

XNOR		
x	y	(x©y)'
0	0	1
0	1	0
1	0	0
1	1	1

Boolean Rules – Based on Axioms, Theorems, and Identities.

Associative rule	a) $x + (y + z) = (x + y) + z$ b) $x(y.z) = (x.y)z$
Commutative rule	a) $x + y = y + x$ b) $x.y = y.x$
Distributive rule	a) $x + (y.z) = (x + y)(x + z)$ b) $x(y + z) = x.y + x.z$
Identity rule	a) $x + 0 = x$ b) $x.1 = x$

Complement rule	a) $x + x' = 1$ b) $x.x' = 0$
Idempotence	a) $x + x = x$ b) $x.x = x$
Absorption	a) $x + (x.y) = x$ b) $x\ (x + y) = x$ c) $x\ (x' + y) = x.y$ $x'\ (x + y) = x'.y$ d) $x + (x'.y) = x + y$ $x' + (x.y) = x' + y$
Null (Zero) or Boundedness	a) $x + 1 = 1$ b) $x.0 = 0$
De Morgan	a) $(x + y)' = x'.y'$ b) $(x.y)' = x' + y'$
Involution (Double negative)	a) $(x')' = x$

Sum of Products – representation of a function of 2^n terms in which each term is a *minterm*.

Minterm - product (ANDing) of all variables in a function, in which each variable appears only once.

m-notation - m_0 m_1 m_2 etc

Karnaugh map – A graphic representation of a Boolean function where squares are used to represent minterms. If a particular square contains a '1', it means that the minterm represented by that square is included in the function.

	B'	B
A'	m_0	m_1
A	m_2	m_3

2 variable Karnaugh map

	B'C'	B'C	BC	BC'
A'	m_0	m_1	m_3	m_2
A	m_4	m_5	m_7	m_6

3 variable Karnaugh map

	C'D'	C'D	CD	CD'
A'B'	m_0	m_1	m_3	m_2
A'B	m_4	m_5	m_7	m_6
AB	m_{12}	m_{13}	m_{15}	m_{14}
AB'	m_8	m_9	m_{11}	m_{10}

4 variable Karnaugh map

The map “wraps” both horizontally and vertically.

Groups -
- Group adjacent squares into groups of 8, 4, 2, or 1.
- Make the groups as large as possible.
- Each new group must include one or more minterms that were not grouped before.
- Wrap around – squares on opposite edges are adjacent.
- A minterm may be included in more than 1 group.

Logic Circuits

A computer is built out of standard components that are collectively referred to as *logic circuits*.
There are 2 categories of logic circuits -

Combinational circuits – made up of a combination of logic gates with n inputs and m outputs. Each output depends on all given inputs. Combinational circuit is memoryless, it does not remember its previous output.

- **Half adder** – can only add 2 bits.
- **Multiplexer** – combinational circuit with n inputs and only one output.

Sequential circuits – includes the concept of memory in the logic. The memory enables the circuit to remember its current state to be used in the future: the future state can be dependent on the current state.

- **Flip-flops** – a storage element that can hold 1 bit. A set of flip-flops can be used to hold a set of bits.
  - **SR flip-flops** – simplest type of flip-flop. The 2 outputs are complements of each other. Output is undefined if both inputs are 1.
  - **D flip-flop –** Output is the same as input. Output remains as is until new input is given. (D stands for data).

- **JK flip-flop –** Removes the undefined state from SR flip-flops. (Invented by Jack Kilby, who invented integrated circuits).

- **T flip-flop** – Input toggles the state of the flip-flop. If input is 0, the next state is the same as the current state. If the input is 1, the next state is the complement of the current state. (T stands for toggle).

- **Synchronous v Asynchronous** – The flip-flops are all *asynchronous* devices: the transition from one state to another can happen only when there is a change in the input.

  Digital computers are *synchronous* devices: A central clock controls the timing of all logic circuits. The clock creates a signal that co-ordinates all events. A simple event takes place only at the 'tick' of the clock signal.

- **Register** – a synchronous (clocked) sequential circuit. An n-bit storage device that stores its data between consecutive clock pulses. At the trigger of the clock, the old data is discarded and replaced by the new data.

- **Digital counter** – an n-bit counter can be made out of n T flip-flops. An n-bit digital counter counts from 0 to $2^n - 1$. Eg: when n = 4, the counter counts from 0 to 15.

--ooOoo--