Computing Theory

Study Guide

Table of Contents

Welcome

Welcome to an exciting journey of discovery and learning at the Academic Institute of Excellence (AIE). AIE is a revolutionised family of brands, people, and students, offering a curriculum modelled on workplace scenarios to provide you with a practical understanding of how to apply your skills in the real world.

At AIE, we aim to provide innovative, quality education, coupled with excellent service delivery and a modern outlook on learning technology. We take pride in developing and supporting students through innovative programs that create well-rounded, employable, and professionally developed individuals. Our learning paths are meticulously designed based on global skill demands and the needs of modern students. Our facilitators are qualified subject matter experts with practical industry experience and tertiary education expertise.

Conveniently located in Greenside, Midrand (Johannesburg), and De Waterkant (Cape Town), our campuses offer a range of recreational possibilities and ample workspace, including computer studios, libraries, and a fully operational Makers Lab. You'll have access to lecturers, resources, and the latest software to help you fulfill your course requirements.

At AIE, we are committed to delivering demand-driven education built upon the principles of quality education, innovation, and technology. Each of our programs is benchmarked against local and international standards to ensure you receive the highest level of quality education.

Join us as we strive towards excellence and empower future generations to become problem solvers, critical thinkers, and innovators—the leaders of tomorrow.

Our VISION

To deliver demand-driven education, built upon the principle of quality education through innovation and technology.

#Flexibility #AnywhereAnytime #Accessibility #StudentSupport

Computing Theory COTH

Module Purpose

The essence of this module is to cultivate a robust understanding of programming concepts and the application of different programming languages in solving problems and performing tasks on computers. As programming forms the backbone of software development and technological innovation, acquiring these skills is vital for anyone looking to make a mark in the IT sector or any field involving computational tasks.

Specific Outcome	Assessment Criteria
KM-04-KT01 - Introduction to Programming Languages	Gain an overview of the diverse programming languages available, understanding their syntax, paradigms, and the specific contexts in which they are most effectively utilized
KM-04-KT02 - Programming Basics	Master the foundational concepts of programming, including variables, data types, control structures, functions, and error handling, setting the stage for more advanced programming challenges.
KM-04-KT03 - Software Applications	Explore how programming languages are applied in the development of software applications, covering a range of domains from web development to mobile apps and system software.

COMPUTER Requirements

It is advisable that students make use of their own personal computers to complete this module.

READ – SAQA Qualification Detail

This module forms part of the Occupational Certificate: Data Science Practitioner Occupational Certificate 118708, NQF 5 Follow the link on AMI to view the full details of the qualification.

Read: Data Science Practitioner SAQA Qualification

Link to SAQA document: Click here

Introduction to Programming Languages

Unit 1

Unit Overview

The following topics are covered in this unit:

- Concepts, principles and terminology
- Developing structured and creative thinking skills through programming
- The logic of programming
- Choosing a programming language
- Syntax
- Learning code
- Installation and set-up

Learning Outcomes

At the end of this unit the student should be able to ....

- Gain an overview of the diverse programming languages available, understanding their syntax, paradigms, and the specific contexts in which they are most effectively utilized.

1.1. Welcome to Problem solving skills for IT Professional

Welcome in this unit, you will explore key concepts and practical applications.

To make the most of your learning experience, follow these instructions:

- Review: Start by reading through the provided materials for this unit. Pay attention to the key ideas and concepts presented.
- Study your prescribed material
- Follow the Study Material References: Utilise the references to the prescribed book(s) to delve deeper into the subject matter. These study materials will enhance your understanding and provide insights.

Feel free to engage in discussions, ask questions, and collaborate with your peers on the StudentHub to deepen your understanding of this unit.

Enjoy your learning journey!

1.1.1. Introduction to this Unit

Programming languages are the building blocks of software development, offering a way to communicate instructions to a computer. They encompass a range of concepts such as syntax, which defines the structure of code, and semantics, which conveys its meaning. Learning a programming language involves mastering these concepts, as well as understanding programming paradigms like procedural, object-oriented, and functional programming. The logic of programming is essential, as it enables developers to break down complex problems into smaller, manageable tasks, fostering both structured and creative thinking. This dual approach not only aids in problem-solving but also encourages innovation in the design and optimization of code.

When starting out, choosing the right programming language is crucial, as different languages are tailored to specific tasks—Python is often preferred for data science, while JavaScript is key in web development. Learning to code involves a gradual process of mastering syntax, practicing problem-solving, and continuously building on new skills. Setting up the development environment, including installing the necessary tools and libraries, is an essential first step to ensure a smooth programming experience. This structured approach lays a solid foundation for any aspiring programmer.

Prev
Welcome
Unit Name 1
Unit Name 2
Unit Name 3
Unit Name 4
Unit Name 5
Next

1.2. KT0101 – Concepts, principles and terminology

A programming language is any set of rules that converts strings, or graphical program elements in the case of visual programming languages, to various kinds of machine code output. Programming languages are one kind of computer language, and are used in computer programming to implement algorithms.

Most programming languages consist of instructions for computers. There are programmable machines that use a set of specific instructions, rather than general programming languages. Since the early 1800s, programs have been used to direct the behavior of machines such as Jacquard looms, music boxes and player pianos.[1] The programs for these machines (such as a player piano's scrolls) did not produce different behavior in response to different inputs or conditions.

Thousands of different programming languages have been created, and more are being created every year. Many programming languages are written in an imperative form (i.e., as a sequence of operations to perform) while other languages use the declarative form (i.e. the desired result is specified, not how to achieve it).

The description of a programming language is usually split into the two components of syntax (form) and semantics (meaning), which are usually defined by a formal language. Some languages are defined by a specification document (for example, the C programming language is specified by an ISO Standard) while other languages (such as Perl) have a dominant implementation that is treated as a reference. Some languages have both, with the basic language defined by a standard and extensions taken from the dominant implementation being common.

Javascript

JavaScript is a high-level programming language that is one of the core technologies of the World Wide Web. It is used as a client-side programming language by 97.8 percent of all websites. JavaScript was originally used only to develop web browsers, but they are now used for server-side website deployments and non-web browser applications as well.

Watch video

Video: JavaScript in 100 Seconds

[JavaScript in 100 Seconds (youtube.com)]

Python

Python is one of the most popular programming languages today and is easy for beginners to learn because of its readability. It is a free, open-source programming language with extensive support modules and community development, easy integration with web services, user-friendly data structures, and GUI-based desktop applications. It is a popular programming language for machine learning and deep learning applications.

Watch video

Video: Python in 100 Seconds

[Python in 100 Seconds (youtube.com)]

Go

Go was developed by Google in 2007 for APIs and web applications. Go has recently become one of the fastest-growing programming languages due to its simplicity, as well as its ability to handle multicore and networked systems and massive codebases.

Go, also known as Golang, was created to meet the needs of programmers working on large projects. It has gained popularity among many large IT companies thanks to its simple and modern structure and syntax familiarity. Companies using Go as their programming language include Google, Uber, Twitch, Dropbox, among many others. Go is also gaining in popularity among data scientists because of its agility and performance.

Watch video

Video: Go in 100 Seconds

[Go in 100 Seconds (youtube.com)]

Java

Java is one of the most popular programming languages used today.

Owned by Oracle Corporation, this general-purpose programming language with its object-oriented structure has become a standard for applications that can be used regardless of platform (e.g., Mac, Windows, Android, iOS, etc.) because of its Write Once, Run Anywhere (WORA) capabilities. As a result, Java is recognized for its portability across platforms, from mainframe data centers to smartphones. Today there are more than 3 billion devices running applications built with Java.

Watch video

Video: JAVA in 100 Seconds

[Java in 100 Seconds (youtube.com)]

1.3. KT0102 - Developing structured and creative thinking skills through programming

What are the 5 creative thinking skills?

Some of the best examples of creative thinking skills may include: lateral-thinking, visual reading, out-of-the-box thinking, copywriting, artistic creativity, problem-solving, analytical mind, and divergent thinking.

What is creative thinking?

Watch video

Video: Creative Thinking: How to Increase the Dots to Connect

[Creative Thinking: How to Increase the Dots to Connect (youtube.com)]

Creative thinking refers to using abilities and soft skills to come up with new solutions to problems. Creative thinking skills are techniques used to look at the issue from different and creative angles, using the right tools to assess it and develop a plan.

The focus on creativity and innovation is important because most problems might require approaches that have never been created or tried before. It is a highly valued skill to have individually and one that businesses should always aspire to have among their ranks. After all, the word creativity means a phenomenon where something new is created.

Creative thinking is a skill and, like any other, it needs constant exercise to stay sharp. You need to regularly expose yourself to situations in which a new idea is needed and surround yourself with like-minded people to achieve this goal.

Such a process is made easier with the use of certain techniques. They help get you on the right mindset and provide the basic structure to reach new ideas on demand.

1.4. KT0103 - The logic of programming

Programming logic is a set of principles that delineates how elements should be arranged so a computer can perform specific tasks. Logical thinking, whether programming or formal, means applying principles in a disciplined manner to achieve an acceptable result.
Logic programming is a programming paradigm which is largely based on formal logic. Any program written in a logic programming language is a set of sentences in logical form, expressing facts and rules about some problem domain. Major logic programming language families include Prolog, answer set programming (ASP) and Datalog. In all of these languages, rules are written in the form of clauses:

Understanding Logic

Logic is human reasoning that is conducted according to strict principles or rules. Programming logic is a set of principles that delineates how elements should be arranged so a computer can perform specific tasks. Logical thinking, whether programming or formal, means applying principles in a disciplined manner to achieve an acceptable result.

Logical thinking skills use progressive analyses to ascertain the available options and to decide on the best solution after weighing all pros and cons. Further, logical thinking is a skill that improves with use. The more opportunities you have to apply those skills the better your critical thinking skills become. That applies not only to logical thinking in general but also to logical thinking in computer science.

Practice is the only way to develop programming logic skills. Logic skills are always evolving as new features are added to programming languages or new development challenges appear. Programming logic is especially crucial in areas such as artificial intelligence and machine learning where the code is attempting to emulate the human thought process.

That being said, visualizing the logical relationships can help you see what the connections are among the different data points. Although there are plenty of programs to help with visualization, paper and pencil is often the better choice. There's something about the physical act of drawing that helps keep a mind focused. Don't worry about notations or conventions. This is a diagram for you to use as a way to develop your skills.

1.5. KT0104 Choosing a programming language

Questions to Ask When Choosing a Programming Language. Does the language have proper ecosystem support?

- Type of Application.
- Targeted Platform.
- Maintainability. ...
- Scalability and Performance.
- Security.
- Community Support. ...
- Development Time Limit.

1.6. KT0105 - Syntax

Watch video

Video: What is Syntax in Computer Programming?
[What is Syntax in Computer Programming? | Basic Syntax for Beginners - KnowledgeHut (youtube.com)]

In computer science, the syntax of a computer language is the set of rules that defines the combinations of symbols that are considered to be correctly structured statements or expressions in that language. This applies both to programming languages, where the document represents source code, and to markup languages, where the document represents data.

The syntax of a language defines its surface form. Text-based computer languages are based on sequences of characters, while visual programming languages are based on the spatial layout and connections between symbols (which may be textual or graphical). Documents that are syntactically invalid are said to have a syntax error. When designing the syntax of a language, a designer might start by writing down examples of both legal and illegal strings, before trying to figure out the general rules from these examples.

Syntax therefore refers to the form of the code, and is contrasted with semantics – the meaning. In processing computer languages, semantic processing generally comes after syntactic processing; however, in some cases, semantic processing is necessary for complete syntactic analysis, and these are done together or concurrently. In a compiler, the syntactic analysis comprises the frontend, while the semantic analysis comprises the backend (and middle end, if this phase is distinguished).

1.7. KT0106 - Learning code

How to Start Coding?

- Take online courses.
- Watch video tutorials.
- Read books and ebooks.
- Complete coding projects.
- Find a mentor and a community.
- Consider enrolling in a coding bootcamp.

1.8. KT0107 - Installation and set-up

Installation (or **setup**) of a computer program (including device drivers and plugins), is the act of making the program ready for execution. Installation refers to the particular configuration of a software or hardware with a view to making it usable with the computer. A soft or digital copy of the piece of software (program) is needed to install it. There are different processes of installing a piece of software (program). Because the process varies for each program and each computer, programs (including operating systems) often come with an installer, a specialised program responsible for doing whatever is needed (see below) for the installation. Installation may be part of a larger software deployment process.

Installation typically involves code (program) being copied/generated from the installation files to new files on the local computer for easier access by the operating system, creating necessary directories, registering environment variables, providing separate program for un-installation etc. Because code is generally copied/generated in multiple locations, uninstallation usually involves more than just erasing the program folder. For example, registry files and other system code may need to be modified or deleted for a complete uninstallation.

Programming basics

Unit 2

Unit Overview

The following topics are covered in this unit:

- Programming environment
- Algorithms
- Data types
- Variables
- Keywords
- Logical and arithmetical operators
- Logical operations
- Loops
- Numbers, characters and arrays
- Function

Learning Outcomes

At the end of this unit the student should be able to:

- Master the foundational concepts of programming, including variables, data types, control structures, functions, and error handling, setting the stage for more advanced programming challenges.

2.1. Welcome to the Programming basics

Welcome in this unit, you will explore key concepts and practical applications.

To make the most of your learning experience, follow these instructions:

- Review: Start by reading through the provided materials for this unit. Pay attention to the key ideas and concepts presented.
- Study your prescribed material
- Follow the Study Material References: Utilise the references to the prescribed book(s) to delve deeper into the subject matter. These study materials will enhance your understanding and provide insights.

Feel free to engage in discussions, ask questions, and collaborate with your peers on the StudentHub to deepen your understanding of this unit.

Enjoy your learning journey!

2.1.1. Introduction to this Unit

Programming basics form the foundation of any coding journey, starting with understanding the programming environment where code is written, tested, and debugged. This environment typically includes a code editor or Integrated Development Environment (IDE) and the necessary compilers or interpreters to run the code. Central to programming is the concept of algorithms—step-by-step instructions designed to solve specific problems. These algorithms are implemented using various data types (like integers, floats, and strings) and variables, which store and manipulate data. Understanding keywords, which are reserved words in a programming language, is also crucial as they form the building blocks for writing valid code.

As you dive deeper, you'll encounter logical and arithmetical operators, which perform operations on data and are essential for making decisions and processing information. Logical operations, combined with loops, enable programs to handle repetitive tasks efficiently. Numbers, characters, and arrays are fundamental data structures that allow for the storage and manipulation of different types of data. Functions play a critical role by encapsulating code into reusable blocks, making programs more organized and easier to manage. Together, these basics provide the tools necessary to build more complex programs and develop problem-solving skills in programming.

Prev
Welcome
Unit Name 1
Unit Name 2
Unit Name 3
Unit Name 4
Unit Name 5
Next

2.2. KT0201 - Programming environment

In a general sense, a programming environment combines hardware and software that allows a developer to build applications.

Developers typically work in integrated development environments or IDEs. These connect users with all the features necessary to write and test their code correctly. Different IDEs will offer other capabilities and advantages.

Examples of Programming Environments

Being familiar with popular programming environment options will help you make an informed decision for your app-building endeavors. Four of the most common IDEs are IntelliJ, Eclipse, NetBeans, and Visual Studio.

Visual Studio

Visual Studio is an IDE developed by Microsoft. It offers a great deal of versatility, providing users with an expansive library of extensions that allows for more customization than other environments. Compatibility testing in Visual Studio is also difficult to match.

2.3. KT0202 - Algorithms

What are basic algorithms?

Watch video

Video: Algorithms Explained for Beginners

[Algorithms Explained for Beginners - How I Wish I Was Taught (youtube.com)]

Algorithm Basics. The word Algorithm means "a process or set of rules to be followed in calculations or other problem-solving operations". Therefore Algorithm refers to a set of rules/instructions that step-by-step define how a work is to be executed upon in order to get the expected results

What are the 4 algorithms?

Algorithm types we will consider include:

- Simple recursive algorithms.
- Backtracking algorithms.
- Divide and conquer algorithms.
- Dynamic programming algorithms.
- Greedy algorithms.
- Branch and bound algorithms.
- Brute force algorithms.
- Randomized algorithms.

2.4. KT0203 - Data types

In computer science and computer programming, a data type or simply type is an attribute of data which tells the compiler or interpreter how the programmer intends to use the data. Most programming languages support basic data types of integer numbers (of varying sizes), floating-point numbers (which approximate real numbers), characters and Booleans. A data type constrains the values that an expression, such as a variable or a function, might take. This data type defines the operations that can be done on the data, the meaning of the data, and the way values of that type can be stored. A data type provides a set of values from which an expression (i.e. variable, function, etc.) may take its values

What are the 5 data types?

Most modern computer languages recognize five basic categories of data types: Integral, Floating Point, Character, Character String, and composite types, with various specific subtypes defined within each broad category.

Watch video

Video: What is Data Type and Its Types?

[What is Data Type and Its Types? | Programming Tutorial for Beginners - KnowledgeHut (youtube.com)]

Prev
Welcome
Unit Name 1
Unit Name 2
Unit Name 3
Unit Name 4
Unit Name 5
Next

2.5. KT0204 - Variables

What are Variables?

Scientists try to figure out how the natural world works. In doing so, they use experiments to search for cause and effect relationships. Cause and effect relationships explain why things happen and allow you to reliably predict what will happen if you do something. In other words, scientists design an experiment so that they can observe or measure if changes to one thing cause something else to vary in a repeatable way.

The things that are changing in an experiment are called variables. A variable is any factor, trait, or condition that can exist in differing amounts or types. An experiment usually has three kinds of variables: independent, dependent, and controlled.

Watch video

Video: What is a Variable?

[What is a Variable? | Variables in Math Introduction | Algebra (youtube.com)]

2.6. KT0205 - Keywords

What are keywords examples?

Keywords are the words and phrases that people type into search engines to find what they're looking for. For example, if you were looking to buy a new jacket, you might type something like “mens leather jacket” into Google. Even though that phrase consists of more than one word, it's still a keyword.

Keywords are predefined, reserved words used in programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier. For example: int money; Here, int is a keyword that indicates money is a variable of type int (integer).

Arithmetic Operators

Definition

The arithmetic operators perform addition, subtraction, multiplication, division, exponentiation, and modulus operations.

Addition	**+**	**Adds one operand to the other**
Subtraction	-	Subtracts the second operand from the first
Multiplication	*	Multiplies one operand by the other
Division	/	Divides the first operand by the second
Modulo	%	Divides the first INTEGER operand by the second, and returns the remainder
Exponentiation	**	Lets you refer to a number in terms of a base value and an exponent

2.7. KT0206 - Logical and arithmetical operators

Arithmetical operators are functions that take numbers as arguments and map onto a new number. Logical operators are functions that take propositions (or whatever that can have only two different values, such as 'true' or 'false') and map them onto either one of those values.

What are logical operators and precedence?

Also like arithmetic operators, logical operators have precedence that determines how things are grouped in the absence of parentheses. In an expression, the operator with the highest precedence is grouped with its operand(s) first, then the next highest operator will be grouped with its operands, and so on.

2.8. KT0207 - Logical operations: if-statements, where-statements, If else conditions

What are logical operations?

A logical operation is a special symbol or word that connects two or more phrases of information. It is most often used to test whether a certain relationship between the phrases is true or false

A logical operation is a special symbol or word that connects two or more phrases of information. It is most often used to test whether a certain relationship between the phrases is true or false.

In computing, logical operations are necessary because they model the way that information flows through electrical circuits, such as those inside a CPU. These types of operations are called boolean operations.

The elements in a circuit which behave according to Boolean logic are called logic gates.

2.9. KT0208 - Loops

What is a loop?

In computer programming, a loop is a sequence of instruction s that is continually repeated until a certain condition is reached. Typically, a certain process is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter has reached a prescribed number. If it hasn't, the next instruction in the sequence is an instruction to return to the first instruction in the sequence and repeat the sequence. If the condition has been reached, the next instruction "falls through" to the next sequential instruction or branches outside the loop. A loop is a fundamental programming idea that is commonly used in writing programs.

An infinite loop is one that lacks a functioning exit routine . The result is that the loop repeats continually until the operating system senses it and terminates the program with an error or until some other event occurs (such as having the program automatically terminate after a certain duration of time).

Watch video

Video: Intro to Programming: Loops

[Intro to Programming: Loops - YouTube]()

2.10. KT0209 - Numbers, characters and arrays

What are character arrays?

A character array is a sequence of characters, just as a numeric array is a sequence of numbers. A typical use is to store short pieces of text as character vectors, such as c = 'Hello World' . A string array is a container for pieces of text. String arrays provide a set of functions for working with text as data.

Watch video

Video: Character arrays and pointers

[Character arrays and pointers - part 1 - YouTube]()

What is array of numbers?

An array is created using square brackets ([and]). For example, an array of numbers can be created as follows: let arr: number[] = []; To create an array of values, you simply add square brackets following the type of the values that will be stored in the array: number[] is the type for a number array.

2.11. KT0210 - Functions

What defines function?

function, in mathematics, an expression, rule, or law that defines a relationship between one variable (the independent variable) and another variable (the dependent variable). Functions are ubiquitous in mathematics and are essential for formulating physical relationships in the sciences.

Prev
Welcome
Unit Name 1
Unit Name 2
Unit Name 3
Unit Name 4
Unit Name 5
Next

2.12. KT0211 - Input and output operations

What is input and output with examples?

An input is data that a computer receives. An output is data that a computer sends. Computers only work with digital information. Any input that a computer receives must be digitised. Often data has to be converted back to an analogue format when it's output, for example the sound from a computer's speakers.

In computing, input/output (I/O, or informally io or IO) is the communication between an information processing system, such as a computer, and the outside world, possibly a human or another information processing system. Inputs are the signals or data received by the system and outputs are the signals or data sent from it. The term can also be used as part of an action; to "perform I/O" is to perform an input or output operation.

I/O devices are the pieces of hardware used by a human (or other system) to communicate with a computer. For instance, a keyboard or computer mouse is an input device for a computer, while monitors and printers are output devices. Devices for communication between computers, such as modems and network cards, typically perform both input and output operations.

Software applications

Unit 3

Unit Overview

The following topics are covered in this unit:

- Basic programming knowledge on HTML, JavaScript (or any scripting language)
- Software development, e.g. C#, C++, Java, .NET, Python, VB
- Databases (SQL)
- Web development technologies
- AI and Machine Learning concept and principles
- Project management methodology (e.g. Agile)
- Tools and technologies including at least one: including SQL and a minimum of one additional language of choice e.g. Python or C# or Visual Basic VB or Java)

Assessment Criteria

At the end of this unit the student should be able to:

- Explore how programming languages are applied in the development of software applications, covering a range of domains from web development to mobile apps and system software.

3.1. Welcome to the Software applications Unit

Welcome in this unit, you will explore key concepts and practical applications.

To make the most of your learning experience, follow these instructions:

- Review: Start by reading through the provided materials for this unit. Pay attention to the key ideas and concepts presented.
- Study your prescribed material
- Follow the Study Material References: Utilise the references to the prescribed book(s) to delve deeper into the subject matter. These study materials will enhance your understanding and provide insights.

Feel free to engage in discussions, ask questions, and collaborate with your peers on the StudentHub to deepen your understanding of this unit.

Enjoy your learning journey!

3.1.1. Introduction to this Unit

The Software Applications Unit provides a comprehensive overview of essential programming and development skills, beginning with basic knowledge of HTML and JavaScript (or any other scripting language), which are fundamental for web development. This unit also delves into software development using languages such as C#, C++, Java, .NET, Python, and Visual Basic (VB), each of which plays a critical role in different types of software projects. Understanding how to work with databases, particularly SQL, is another key focus, enabling the development of robust, data-driven applications.

In addition to traditional programming, the unit introduces web development technologies and explores the emerging fields of AI and Machine Learning, covering their core concepts and principles. To effectively manage software projects, the unit also covers project management methodologies, with a particular emphasis on Agile, which is widely used in the industry. Students will also gain hands-on experience with tools and technologies, including SQL and at least one additional programming language of choice, such as Python, C#, or Java, ensuring they are well-equipped to handle various aspects of software development.

3.1. KT0301 - Basic programming knowledge on HTML, JavaScript (or any scripting language)

Watch video

Video: HTML in 100 Seconds

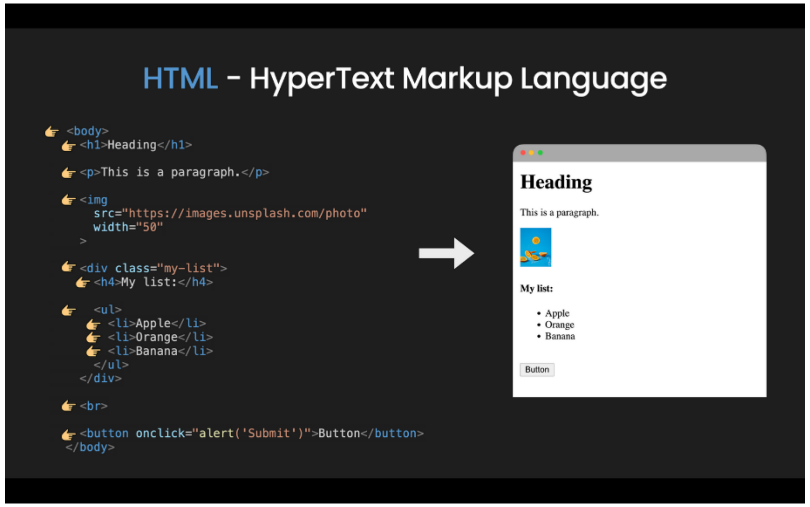
[HTML in 100 Seconds (youtube.com)]

What is basic knowledge of HTML?

HTML (HyperText Markup Language) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables

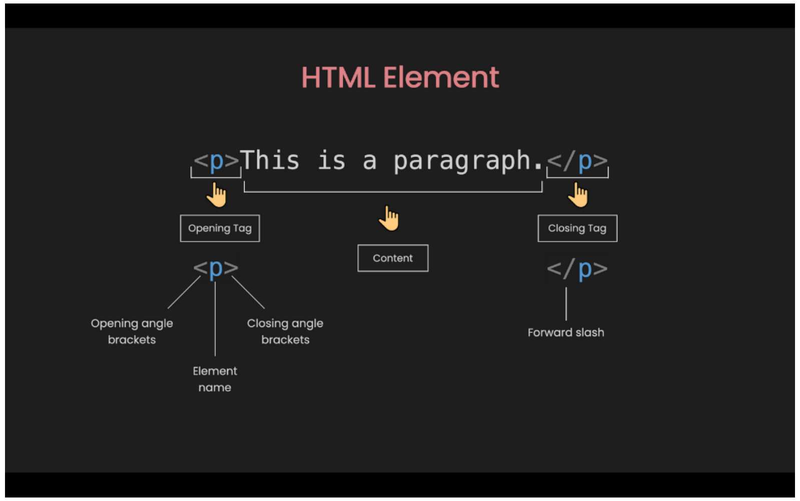
What Is HTML?

HTML, which stands for Hypertext Markup Language, is a pretty simple language. It consists of different elements which we use to structure a web page.



```
<body>
  <h1>Heading</h1>

  <p>This is a paragraph.</p>

  <img
    src="https://images.unsplash.com/photo"
    width="50"
  >

  <div class="my-list">
    <h4>My list:</h4>

    <ul>
      <li>Apple</li>
      <li>Orange</li>
      <li>Banana</li>
    </ul>
  </div>

  <br>

  <button onclick="alert('Submit')">Button</button>
 </body>
```

What Are HTML Elements?



The element usually starts with an opening tag, which consists of the name of the element. It's wrapped in opening and closing angle brackets. The opening tag indicates where the element begins.

Similar to the opening tag, the closing tag is also wrapped in opening and closing angle brackets. But it also includes a forward slash before the element's name.

Everything inside the opening and closing tags is the content.

But not all elements follow this pattern. We call those that don't empty elements. They only consist of a single tag or an opening tag that cannot have any content. These elements are typically used to insert or embed something in the document.

3.2. KT0302 - Software development, e.g. C#, C++, Java, .NET, Python, VBA

What is meant by software development?

Software development refers to a set of computer science activities dedicated to the process of creating, designing, deploying and supporting software. Software itself is the set of instructions or programs that tell a computer what to do. It is independent of hardware and makes computers programmable.

What is software development?

Software development refers to a set of computer science activities dedicated to the process of creating, designing, deploying and supporting software.

Software itself is the set of instructions or programs that tell a computer what to do. It is independent of hardware and makes computers programmable. There are three basic types:

- System software to provide core functions such as operating systems, disk management, utilities, hardware management and other operational necessities.
- Programming software to give programmers tools such as text editors, compilers, linkers, debuggers and other tools to create code.
- Application software (applications or apps) to help users perform tasks. Office productivity suites, data management software, media players and security programs are examples. Applications also refers to web and mobile applications like those used to shop on Amazon.com, socialize with Facebook or post pictures to Instagram.1

A possible fourth type is embedded software. Embedded systems software is used to control machines and devices not typically considered computers — telecommunications networks, cars, industrial robots and more. These devices, and their software, can be connected as part of the Internet of Things (IoT).2

Software development is primarily conducted by programmers, software engineers and software developers. These roles interact and overlap, and the dynamics between them vary greatly across development departments and communities.

Programmers, or coders, write source code to program computers for specific tasks like merging databases, processing online orders, routing communications, conducting searches or displaying text and graphics. Programmers typically interpret instructions from software developers and engineers and use programming languages like C++ or Java to carry them out.

Software engineers apply engineering principles to build software and systems to solve problems. They use modeling language and other tools to devise solutions that can often be applied to problems in a general way, as opposed to merely solving for a specific instance or client. Software engineering solutions adhere to the scientific method and must work in the real world, as with bridges or elevators. Their responsibility has grown as products have become increasingly more intelligent with the addition of microprocessors, sensors and software. Not only are more products relying on software for market differentiation, but their software development must be coordinated with the product's mechanical and electrical development work.

Software developers have a less formal role than engineers and can be closely involved with specific project areas — including writing code. At the same time, they drive the overall software development lifecycle — including working across functional teams to transform requirements into features, managing development teams and processes, and conducting software testing and maintenance.3

The work of software development isn't confined to coders or development teams. Professionals such as scientists, device fabricators and hardware makers also create software code even though they are not primarily software developers. Nor is it confined to traditional information technology industries such as software or semiconductor businesses. In fact, according to the Brookings Institute (link resides outside of ibm.com), those businesses "account for less than half of the companies performing software development."

An important distinction is custom software development as opposed to commercial software development. Custom software development is the process of designing, creating, deploying and maintaining software for a specific set of users, functions or organizations. In contrast, commercial off-the-shelf software (COTS) is designed for a broad set of requirements, allowing it to be packaged and commercially marketed and distributed.

3.3. KT0303 - Databases (SQL)

Watch video

Video: SQL Explained in 100 Seconds

SQL Explained in 100 Seconds - YouTube

What are databases in SQL?

A database in SQL Server is made up of a collection of tables that stores a specific set of structured data. A table contains a collection of rows, also referred to as records or tuples, and columns, also referred to as attributes.

Types of database management systems

- hierarchical database systems.
- network database systems.
- object-oriented database systems.

3.4. KT0304 - Web development technologies

What are web technologies?

Web technologies refers to the way computers/devices communicate. with each other using mark up languages. It invo lt is communication. across the web, and create, deliver or manage web content using hypertext markup language (HTML).

Top Web App Development Technologies

1. WebAssembly

   WebAssembly is another huge accomplishment in the web improvement world. This is viewed as the closest companion of JavaScript with some announcing the up and coming age of Javascript which is a little and quick parallel configuration that guarantees close local execution for web applications. WebAssembly can assist designers with building rapid web applications that are especially required in games, music, CAD applications, video altering, and transferring.

2. Movement User Interface (UI) Design

   Clients today request remarkable client experience, a better UI plan that is simple than use with natural safe progression of data is the mystery of drawing in target crowds. Site pages created with appealing and intuitive components and intriguing page format will have clients running to website pages and convey expanded harp time on site pages. Innovative enlivened headers and pennants, vivified outlines, foundation liveliness, and drift impacts offer life to pages and draw in clients are some of the best technology to build website.

3. Utilization of Chatbots

   In the event that you embrace a Chatbot, at that point, it will upgrade your online help by giving more accommodation to your clients. As per the NewVoiceMedia research report, organizations bear the loss of almost $62 million consistently in light of badly arranged client administrations in the USA. Chatbots are generally trusted to fulfill client care needs well indeed. Furthermore, you will likewise be profited by the day in and day out help. Doing as such, you can spare the compensation of a full-time online client assistance agent and spend that on something attainable.

4. Artificial Intelligence

   As per Gartner, man-made brainpower (AI) is estimated to reach $3.9 trillion out of 2022. Numerous associations are fusing AI into their advanced change technique, and there's no indication of it easing back down. As of late, Babylon by TELUS Health made a free social insurance versatile application that permits you to check manifestations, talk with specialists, and access your wellbeing records.

3.5. KT0305 - AI and Machine Learning concept and principles

What are the principles of machine learning?

The three components that make a machine learning model are representation, evaluation, and optimization. These three are most directly related to supervised learning, but it can be related to unsupervised learning as well. Representation - this describes how you want to look at your data.

Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.[2] Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers; but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytics.

How are deep learning and machine learning related?

Machine learning is the broader category of algorithms that are able to take a data set and use it to identify patterns, discover insights and/or make predictions. Deep learning is a particular branch of machine learning that takes ML's functionality and moves beyond its capabilities.

With machine learning in general, there is some human involvement in that engineers are able to review an algorithm's results and make adjustments to it based on their accuracy. Deep learning doesn't rely on this review. Instead, a deep learning algorithm uses its own neural network to check the accuracy of its results and then learn from them.

A deep learning algorithm's neural network is a structure of algorithms that are layered to replicate the structure of the human brain. Accordingly, the neural network learns how to get better at a task over time without engineers providing it with feedback.

The two major stages of a neural network's development are training and inference. Training is the initial stage in which the deep learning algorithm is provided with a data set and tasked with interpreting what that data set represents. Engineers then provide the neural network with feedback about the accuracy of its interpretation, and it adjusts accordingly. There may be many iterations of this process. Inference is when the neural network is deployed and is able to take a data set it has never seen before and make accurate predictions about what it represents.

3.6. KT0306 - Project management methodology (e.g. Agile)

What are project management methodologies?

The project management methodologies list

- Waterfall methodology. The Waterfall method is a traditional approach to project management
- Agile methodology.
- Scrum methodology.
- Kanban methodology.
- Scrumban methodology.
- eXtreme programming (XP) methodology.
- Adaptive project framework (APF) methodology.
- Lean methodology.

3.7. KT0307 - Tools and technologies including at least one: including SQL and a minimum of one additional language of choice e.g. Python or C# or Visual Basic VB or Java)

What is tools and technology?

The Tools & Technology Finder allows you to learn about the machines, equipment, tools and software that workers commonly use in specific occupations. Job seekers can identify new technical skills, competencies, and cutting-edge technologies used in today's information and technology-driven economy.

Tools and technologies

- Product planning and innovation. Quality planning: Software-QFD. ...
- Software analysis, arhitecture and design. Brainstorming. Mind-Map. ...
- Project management. Project planning and management. Jira, Confluence, MS Project, ScrumDesk. ...
- Testing. Testing tools and frameworks. ...
- UX/UI Design. UX Tools