Syntax : is the rules for how a sentences is constructed

Semantic : the actyual meaning of statements

Question 1: Why do we need to learn the syntax and semantic of a programming language?

* To allow us to clearly express what we want the computer to do.

Script : a program that’s short, simple, and can be written very quickly

What is Automation?

* The process of replacing a manual step with one that happens automatically. Or Automation is used to replace a repetitive manual step with one that happens automatically.

But, automation is not the solution for every conditiopn. Automation will very usefull is it’s worthy from cost, and the effort.

Question 2 : Whats Automation?

* The process of replacing a manual step with one that happens automatically.

**Note :** Automation is powerfull tool when used in the right place at the right moment.

**Appropriate uses for automation include:**

* The automatic timing and regulation of traffic lights
* A repetitive task that is at high risk for human error
* Sending commands to a computer
* Detecting and removing duplicates of data
* Sending automated emails that are personalized by pulling individual names from a database and plugging them into the email
* Updating a large number of file permissions
* Reporting on system data, like disk or memory usage
* Installing software
* Generating reports
* Deploying a file or a computer program to all computers on a company network
* Using a configuration management system to deploy software patches, after a human has designed the system
* Populating an e-commerce site with products
* Setting the home directory and access permissions for users

**Automation is not always an appropriate or complete solution**

Automation cannot perform all human work. Tasks that call for human creativity, social connection, psychology, flexibility, ingenuity, evaluation, and/or complex analytic work are not good candidates for full automation. The following are some examples of tasks that cannot or should not be fully automated:

* Items that require human evaluation and analytic skills:
  + Designing a configuration management system
  + Investigating and troubleshooting all end user problems
  + Writing a computer program
  + Building a new startup business
* Items that require human creativity and/or an eye for aesthetic qualities:
  + Designing an attractive webpage (AI can do this, but simple automation cannot)
  + Wedding photography
  + Haircuts and styling
* Items that cannot be automated due to basic physics:
  + Troubleshooting or repairing machines that cannot power on or boot up
* Items that need human interaction, psychology, and/or evaluation skills:
  + Interviewing and hiring new employees
  + Customer service (chat bots cannot address every customer service need)
* Items that should not be fully automated due to costs and safety:
  + Grocery store checkout process, including bagging groceries
  + Tasks that are less expensive to perform manually

**Note :** It is important to understand that basic automation is not the same as artificial intelligence. Automation is used to explicitly instruct a machine on how to perform a task. Artificial intelligence (AI) involves training a computing machine to perform more complex tasks through a process called machine learning. This process prepares the AI software to perform new tasks without a human needing to program explicit instructions for each task. Although AI is often used for automating human tasks, AI automation is much more complex than basic automation.

**Key Terms**

**Programming code** - Programming code is a set of written computer instructions, guided by rules, using a computer programming language. It might help to think of the computer instructions as a detailed, step-by-step recipe for performing tasks. The instructions tell computers and machines how to perform an action. Programming code may also be referred to as source code or scripts.

**Programming languages** - Programming languages are similar to human spoken languages in that they both use syntax and semantics. Programming languages are used to write computer programs. Some common programming languages include Python, Java, C, C++, C#, and R.

**Syntax** - Syntax is a set of rules for how statements are constructed in both human and computer languages. Programming syntax includes rules for the order of elements in programming instructions, as well as the use of special characters and their placements in statements. This concept is similar to the syntax rules for grammar and punctuation in human language.

**Semantics** - Semantics refers to the intended meaning or effect of statements, or collections of words, in both human and computer languages. Semantic errors are also referred to as logical errors.

**Computer program** - A computer program is a step-by-step list of instructions that a computer follows to reach an intended goal. It is important to be clear and precise about the actions a computer program is supposed to perform because computers will do exactly what they are instructed to do. Computer programs can be long, complex, and accomplish a variety of tasks. They are often developed by computer programmers and software engineers, but anyone can learn to create them. Computer programs may involve a structured development cycle. They can be written in a wide variety of programming languages, such as Python, Java, C++, R, and more. The completed format of a program is often a single executable file.

**Script** - Scripts are usually shorter and less complex than computer programs. Scripts are often used to automate specific tasks. However, they can be used for complex tasks if needed. Scripts are often written by IT professionals, but anyone can learn to write scripts. Scripts have a shorter, less structured development cycle as compared to the development of complex computer programs and software. Scripts can be written in a variety of programming languages, like Python, Javascript, Ruby, Bash, and more. Some scripting languages are interpreted languages and are only compatible with certain platforms.

**Automation** - Automation is used to replace a repetitive manual step with one that happens automatically.

**Output** - Output is the end result of a task performed by a function or computer program. Output can include a single value, a report, entries into a database, and more.

**Input** - Input is information that is provided to a program by the end user. Input can be text, voice, images, biometrics, and more.

**Functions** - A function is a reusable block of code that performs a specific task.

**Variables** - Variables are used to temporarily store changeable values in programming code.

**Why pick Python?** Well, we chose Python for a few reasons. First off, programming in Python usually feels similar to using a human language. This is because Python makes it easy to express what we want to do with syntax that's easy to read and write. Now I am sure you are wondering what the heck is a Python interpreter. In programming, an interpreter is the program that reads and executes code. **Python interpreter** is the program that reads what is in the recipe and translates it into instructions for your computer to follow.

Python makes it easy to express the fundamental concepts of programming like data structures and algorithms with easy to read syntax. This makes Python a great language to use to learn programming. And there are other reasons to pick Python, too. Python is super popular in the IT industry, making it one of the most common programming languages used today. You can use Python to calculate statistics, run your e-commerce site, process images, interact with web services, and do a whole host of other tasks. Python is perfect for automation. It lets you automate everyday tasks by writing simple scripts that are easy to understand and easy to maintain.

There are also general-purpose scripting languages similar to Python, like Perl or Ruby, which are also widely used for scripting and automation. JavaScript, which was originally developed as a client-side scripting language for the web, is increasingly used server-side for a broader set of tasks. And the list doesn't stop there. There's a vast array of traditional languages to explore like C, C++, Java, or Go.

**Python is:**

* a general purpose scripting language;
* a popular language used to code a variety of applications;
* a frequently used tool for automation;
* a cross-platform compatible language;
* a beginner-friendly language.

**Python is not:**

* a platform-specific / OS-specific scripting language;
* a client-side scripting language;
* a purely object-oriented programming language.

**Key Terms**

**Platform-specific / OS specific scripting language** - Platform-specific scripting languages, like PowerShell (for Windows) and Bash (for Linux), are used by system administrators on those platforms.

**Client-side scripting language** - Client-side scripting languages, like JavaScript, are used mostly for web programming. The scripts are transferred from a web server to the end-user’s internet browser, then executed in the browser.

**Machine language** - Machine language is the lowest-level computer language. It communicates directly with computing machines in binary code (ones and zeros). In binary code, one equals a pulse of electricity and zero equals no electrical pulse. Machine language instructions are made from translating languages like Python into complex patterns of ones and zeros.

**Cross-platform language** - Programming language that is compatible with one or more platforms / operating systems (e.g., Windows, Linux, Mac, iOS, Android).

**Object-oriented programming language** - In object-oriented programming languages, most coding elements are considered to be objects with configurable properties. For example, a form field is an object that can be configured to accept only dates as input in the mm/dd/yy format, and can be configured to read from and write to a specific database.

**Python interpreter** - An interpreter is the program that reads and executes Python code by translating Python code into computer instructions.

**Functions** are pieces of code that perform a unit of work. In the examples you've seen so far, you have only encountered the print() function, which outputs a message to the screen. **Keywords** are reserved words that are used to construct instructions. These words are the core part of the language and can only be used in specific ways.

In the next few weeks, you will also learn the following keywords:

Values: **True, False, None**

Conditions: **if, elif, else**

Logical operators: **and, or, not**

Loops: **for, in, while, break, continue**

Functions: **def, return**

Python actually has a lot more advanced numeric capabilities that are used for data analysis, statistics, machine learning, and other scientific applications. We won't get into these in this course.

**Arithmetic operators**

Python can calculate numbers using common mathematical operators, along with some special operators, too:

x + y Addition + operator returns the sum of x plus y

x - y Subtraction - operator returns the difference of x minus y

x \* y Multiplication \* operator returns the product of x times y

x / y Division / operator returns the quotient of x divided by y

x\*\*e Exponent \*\* operator returns the result of raising x to the power of e

x\*\*2 Square expression returns x squared

x\*\*3 Cube expression returns x cubed

x\*\*(1/2) Square root (½) or (0.5) fractional exponent operator returns the square root of x

x // y Floor division operator returns the integer part of the integer division of x by y

x % y Modulo operator returns the remainder part of the integer division of x by y

**Order of operations**

The order of operations are to be calculated from left to right in the following order:

1. Parentheses ( ), { }, [ ]
2. Exponents (x\*\*e)
3. Multiplication \* and Division /
4. Addition + and Subtraction -

You might find the **PEMDAS** mnemonic device to be helpful in remembering the order.