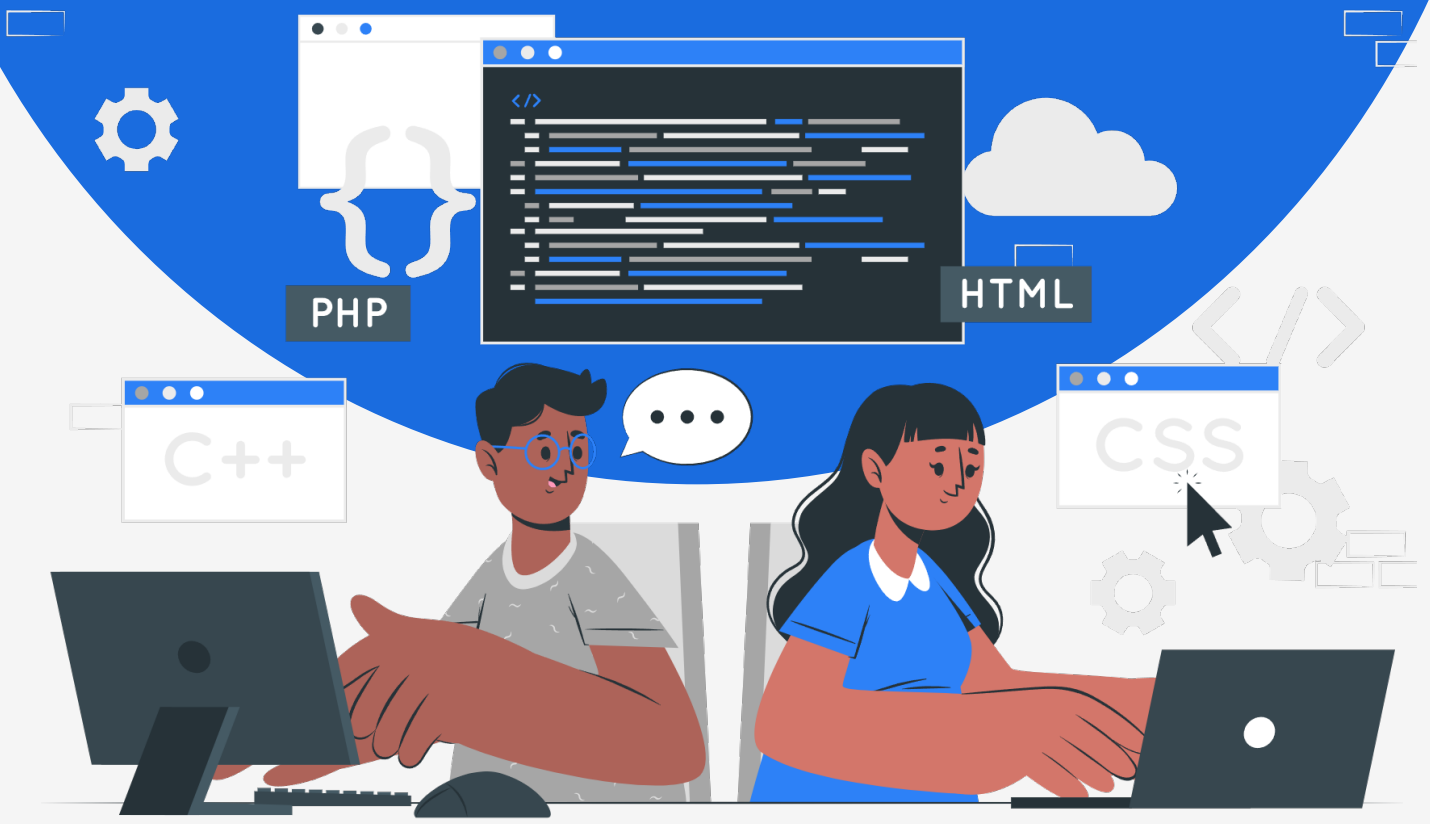
Lesson Plan:

Introduction to Python



PW SKILLS

Topics to be covered:

1. Overview of Programming Languages
2. Python vs. Other Programming Languages
3. What is Pseudocode?
4. Pseudocode Implementation
5. How to write Computer Code Logic?
6. Introduction to Python
7. Why should you learn Python programming?
8. Setting Up Python Environment
9. What can be done with Python?

1. Overview of Programming Languages:

- A programming language is a formal language for creating instructions for a computer to follow. It is a set of rules and symbols that define the process of writing a computer programme. Software programmes, websites, and embedded devices are all developed using programming languages.

- To learn a programming language, you need to understand the basics of syntax, data types, variables, operators, and control structures. You should also learn how to use functions and modules to organize your code. Once you have a good understanding of the basics, you can start to learn about more advanced topics, such as object-oriented programming and design patterns.

Analogy of Programming Languages:

- A programming language is like a set of special rules and symbols that you use to give instructions to a computer.
  - **Example:** Think of it like a recipe. Just like you follow a recipe to cook your favorite dish, a computer follows instructions in a programming language to perform tasks.

- You can use a programming language to create different things, such as:
  - Software programs (like computer games, word processors, or web browsers)
    Websites
  - Embedded devices (like the software inside your smartphone or smartwatch)
  - **Example:** Imagine you want to build a robot. You can use a programming language to tell the robot how to move, what to do when it senses an obstacle, and so on.

To learn a programming language, you start with the basics. You need to understand things like:

- **Syntax:** How to write the instructions in a way the computer understands. It's like following grammar rules in a language.
- **Algorithm:** An algorithm is a well-defined, step-by-step set of instructions or a systematic approach for solving a specific problem or accomplishing a particular task.
- **Data types:** A data type is a classification that specifies which type of value a variable can hold in computer programming i.e. Different kinds of information, like numbers, text, and more.
- **Variables:** A variable is a fundamental concept in computer programming and mathematics, which holds information. For example, a variable as a box that can store numbers, words, or other stuff.
- **Operators:** Operators are symbols or special keywords in programming that perform operations on data or variables or special symbols to do things like math or comparisons.
- **Control Structures:** Control structures are fundamental elements in computer programming that determine the flow of a program, to make decisions and repeat actions in your program(if, else, for, while loops).
- **Functions and modules:** A function is a self-contained and reusable block of code that performs a specific task or set of tasks. Modules are collections of related functions, variables, and other code elements that are bundled together for a specific purpose.

Example: Learning a programming language is a bit like learning a new sport. You start with the basic rules, like how to hold the ball and move your body. Then, you learn more advanced techniques, like strategies and teamwork.

There are many different programming languages, each with its own strengths and weaknesses. Some of the most popular programming languages include:

Python: A general-purpose language noted for its ease of use and readability. It is frequently used in web development, data science, and machine learning.

Java: A general-purpose programming language noted for its portability and security. It is frequently used in the creation of corporate applications and mobile apps.

C/C++: High-performance programming languages that are often used in operating systems, gaming, and embedded devices.

JavaScript: A programming language designed to add interactivity to web pages and to create online apps.

Go: A more recent language that is gaining popularity due to its ease of use and concurrency capabilities.

Programming languages can be classified into different types, based on their characteristics and how they are utilized. Some examples of common programming languages are:

- **General-purpose languages:** These languages can be used to develop a wide variety of software applications. Examples include Python, Java, and C++.
- **Special-purpose languages:** These languages are designed for specific tasks, such as web development, data science, or game development. Examples include JavaScript, R, and Unityscript.
- **Scripting languages:** These languages are used to automate tasks and to add interactivity to web pages. Examples include JavaScript, Python, and Bash.
- **Compiled languages:** These languages are converted into machine code before they can be executed by a computer. Examples include Java, C++, and Go.
- **Interpreted languages:** These languages are executed directly by the computer, without being converted into machine code first. Examples include Python, JavaScript, and PHP.

Learning a programming language might be difficult, but it can also be quite rewarding. Programming is a highly sought-after ability that may lead to a variety of career options. If you want to learn to code, there are several resources accessible to assist you.

2. Python vs. Other Programming Languages:

Python is a popular programming language known for its simplicity and readability. When comparing Python to other programming languages, there are several factors to consider:

2.1. Ease of Learning: Python is frequently suggested for beginners because of its simple syntax. Other languages, such as C++ or Java, may have more difficult learning curves.

2.2. Versatility: Python is a flexible programming language that is utilized in a variety of disciplines such as web development, data analysis, machine learning, and scientific computing. It is not confined to a particular application.

2.3. Readability: Python's code is simple to understand and maintain, making it ideal for collaborative projects.

2.4. Large Standard Library: Python has a comprehensive standard library that simplifies typical programming tasks and eliminates the need to develop code from scratch.

2.5. Community and Ecosystem: Python has a huge and active community, which means there is plenty of help, libraries, and frameworks accessible. This is true for other popular languages as well, but Python's environment is especially robust.

2.6. Cross-Platform: Python is cross-platform, which means you can execute your code on different operating systems without changing anything.

2.7. Scripting Language: Python is frequently used for scripting activities and automation due to its ease of use and rapid development capabilities.

2.8. Integration: Python can be easily integrated with other languages like C and C++ for performance-critical parts of an application.

2.9. Performance: Python is an interpreted language, which can make it slower than compiled languages like C++ in certain situations. However, Python's performance has been improving with the development of JIT (Just-In-Time) compilers like PyPy and libraries like NumPy and Cython for numeric computations.

2.10. Use Cases: Python is a great choice for web development (Django, Flask), data analysis (Pandas, NumPy), and machine learning (TensorFlow, PyTorch). For systems programming, C and Rust are more common. For mobile app development, languages like Swift (iOS) and Kotlin (Android) are preferred.

2.11. Concurrency and Parallelism: Python's Global Interpreter Lock (GIL) can limit its ability to fully utilize multi-core processors for parallel processing. Languages like Go or Rust provide better support for concurrency.

2.12. Security: Python's simplicity and readability can be advantageous for writing secure code. However, security depends more on the developer's practices and awareness.

2.13. Job Opportunities: Python has a strong job market, particularly in data science, web development, and machine learning. The demand for other languages may vary depending on the region and industry.

2.14. Legacy Systems: Some older systems and software rely on languages like COBOL or Fortran, so Python may not be a suitable choice for maintaining or integrating with such systems.

In conclusion, Python is a versatile and beginner-friendly language that excels in many areas, particularly data science and web development. However, the choice of a programming language should be based on the specific needs and goals of your project or application. Different languages have their own strengths and weaknesses, and the best choice depends on the context and requirements.

3. What is Pseudo code?

- Pseudocode is a term which is often used in programming and It is a methodology that allows the programmer to represent the implementation of an algorithm.
- It is a way of writing programs in which you represent the sequence of actions and instructions (aka algorithms) in a form that humans can easily understand.
- Pseudocodes are simple to interpret by everyone irrespective of their programming background.
- Pseudocode, as the name suggests, is a false code or a representation of code which can be understood by even a layman with some school level programming knowledge.

4. Pseudo code Implementation:

Example 1:

Code to find largest among two nos:

```
BEGIN
    NUM num1, num2
    DISPLAY "Enter num1 value: "
    INPUT num1
    DISPLAY "Enter num2 value: "
    INPUT num2

    IF num1 > num2 THEN
        DISPLAY "num1 is larger than num2"
    ELSE
        DISPLAY "num2 is larger than num1"
END
```

Example 2:

Find area of circle:

```
BEGIN
    NUM radius, area, PI
    PI = 22/7
    DISPLAY "ENTER THE RADIUS OF CIRCLE : "
    INPUT radius
    area = PI*radius*radius
    DISPLAY "AREA OF CIRCLE : " area
END
```

5. How to write Computer Code Logic?

Practice writing a lot of code:
Start reading lots of open source code and try to make contributions to open source code.

Check solutions by other people:
It's a good programming practice to read code written by others, it will help you in understanding different approaches for the same problem.

Use a pen and paper to work out solutions:
It is always a good practice to write down the logic of the code on pen and paper, It will help you build strong logic for your programs.

Keep learning new things:
Whenever you find something new just code it out, that's the best way to learn programming.

Be consistent:
Daily Assign some amount of time apart from your regular for just programming. This will keep you up-to-date.

Face problems head-on:
If you have problems early on while learning it's good because it will lay a solid foundation for programming.

6. Introduction to Python:

- Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.
- It was created by Guido van Rossum and first released in 1991.
- It has the following features that makes it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together:
    - High-level built in data structures
    - Dynamic typing
    - Dynamic binding
- Python's simple, easy to learn syntax emphasizes on code readability with the use of whitespaces and therefore reduces the cost of program maintenance.
- Python supports modules and packages, which encourages program modularity and code reuse.
- The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.
- Python is a great general-purpose programming language on its own, but with the help of a few popular libraries (numpy, scipy, matplotlib) it becomes a powerful environment for scientific computing.

7. Why should you learn Python programming?

Among numerous languages available in the market why should you choose python? This is the first question that arises in the mind of new users.

Following are the some of the reasons why people select python:

- **Quality of software:** Python was meant for readability. It's reusable and maintainable as compared to other languages. It's easier to understand. It supports all the modern features like OOPs and functional programming.

- **Productivity of Developers:** The same program which is written in other high-level languages like c++ or java can be written in one-third or one-fifth line of code. That means debugging can be easy and it will be less prone to error which in turn increases the productivity of the developers.

- **Portability:** Mostly it's platform-independent. It can run on any platform or OS with minor or no change at all which makes it a highly portable language. Now you can use Micro Python to interact with hardware as well. It can be used on most of the edge devices.

- **Supporting Libraries:** Python already has a lot of inbuilt libraries that come with the standard python package which you download from its official site. With these libraries, you can build lots of basic applications or day to day automation tasks like copying data in bulk from one place to another. Apart from this, there's a huge list of third-party libraries like Numpy, Matplotlib, Scikit Learn, etc.

- **Fun to use:** Its simplicity and availability of lots of supporting libraries plus huge open source community support make development in python a breeze. That's why its widely preferred by hobbyists as well.

8. What can be done with Python?

- System Programming
- Graphical User Interface
- Web Scraping
- Managing Database
- Fast Prototyping
- Numeric / Scientific Programming
- Game development
- Image Processing
- Robotics
- Automation
- Data science
- Data Mining