NASSCOM® FOUNDATION

Directorate General of Training

Powered by Microsoft

Creating and Running Simple Python Programs

Creating and Running Simple Python Programs

Introduction to Python

What are the uses of Python programming?

What are the basic concepts of Python programming?

What is coding?

What is Coding?

- Coding refers to the process of writing instructions for a computer to perform a task.
- In simpler terms, it is telling the computer what to do.
- Coding involves logic, reasoning, problem-solving skills, organising, focus, persistence and patience.

Advantages of Coding

- Various applications, games and software programs are developed based on coding.
- Coding is used in almost every aspect of life.

Python Programming

what is python programming?

- Python is an easily interpretable, high-level and general-purpose programming language.
    - Freely usable - Developed under an approved open-source license
    - Supports the development of many applications
    - Focuses on the solution rather than language and functionality of the application
    - Helps to work more quickly and integrate systems more effectively

Applications of Python Programming

- Web and internet development
- Scientific and numeric computing
- Education: For teaching programming
- Desktop Graphical User Interfaces or GUIs
- Software development: To build, control, manage and test
- Business applications
- Database access
- Games and 3D graphics

- Network programming

Features of Python Programming

- Easy to read, learn and write programs or codes
- Hosts and compatible with various platforms and systems
- Enables simpler, less-cluttered syntax and grammar
- Uses English keywords
- Adopts test driven development approach

Prerequisites of Python Programming

- Install Python program in the system
- Install standard Python development environment – Python Interpreter
    - There are two ways to use the Python Interpreter. They are interactive mode to execute individual Python statements instantaneously and script mode to create and edit Python source files.
    - It can be written in command line or in the server using the '.py' file extension and then running it in command line.
    - It is well designed and written in fewer lines and relies on Python syntax and Python indentation.

Functions of Python Programming

- The most common function used in Python is print.
- Python’s print() function is used to print information on the screen. The information that has to appear on the screen should be provided in single or double quotes.
- The input() function is used to accept data from the user of the programme.
- The Comments are statements that are used to explain Python code.

Variables of Python Programming

- Variables are names that store data.
- These are created the moment a value is assigned to the name.
- Python does not have any command to declare variables.
- A variable can change the data type even after it is set.

Rules for Python Variables:

- It must start with a letter or the underscore character.
- It can only contain alpha-numeric characters and underscores (A-Z, 0-9, and _).
- It is case-sensitive (Example: age, Age and AGE).
- It cannot start with a number.
- It allows to:
    - Assign values to multiple variables in one line
    - Same value can be assigned to multiple variables

Python Data Types

- In python programming, different types of data can be stored in variables.
- Text Type
    - Str - A sequence of characters, enclosed in single quote or double quotes.
- Numeric Type
    - Int - Integers are whole numbers consisting of '+' or '-' sign with decimal digits like 10, -20.
    - Float - Floating points are the numbers with fractions or decimal points.
    - Complex - Complex numbers are written with a "j" as the imaginary part.
- Sequence Type
    - List - A collection, which is ordered and changeable.
    - Tuple - Tuple is a collection, which is ordered and unchangeable. In Python, tuples are written with round brackets.
    - Range - Used to repeat a set of statements for a specified number of times.

- Mapping Type
  - Dict - A collection, which is unordered, changeable and indexed.
- Set Type
  - Set - A collection, which is unordered and unindexed.
- Boolean Type
  - Bool - Booleans represent either True or False.
- Binary Type

Python Operators

- Arithmetic operators - Used with numeric values to perform common mathematical operations
- Assignment operators - Used to assign values to variables.
- Comparison operators - Used to compare two values.
- Logical operators - Used to combine conditional statements.
- Identity operators - Used to compare the objects.
- Membership operators - Used to test if a sequence is presented in an object.
- Bitwise operators - Used to compare (binary) numbers.

Python Keywords

- Python keywords are used for a predefined purpose.
- These words have a specific meaning for the Interpreter.
- Hence, they should not be used for other purposes.
    - IF - An "if" statement is written by using the 'if' keyword.
    - Elif - If the previous conditions were not true, then, you can give another condition using elif.
    - Else - The else keyword is used to give condition, which is not provided in previous conditions.
    - Nested If - If an 'if' statement is used inside an 'if', then, it is known as Nested If.
    - And / Or - These are logical operators used to combine conditional statements.

Errors in Python

- The errors can be categorised into:
    - Syntax errors - Python program has its own rules of programming. If there is any error in the coding or syntax, it will display syntax error.
    - Logical errors - Python program has its own rules of programming. If there is any error in the coding or syntax, it will display syntax error.
    - Runtime errors - It causes abnormal termination of program while it is being executed. It will appear after the program starts running.

Using Python from Command Line

Questions Discussion

How to install and use of Command line?

What is Google Colab and Azure note book?

Install and Use of Command Line

1. Run Python from the Command Line
2. Run Pip from the Command Line
3. Update Pip, Setuptools, and Wheel
4. Create a Virtual Environment
5. Tools to create a Virtual Environment
6. Use Pip for installing
7. Installing from PyPI
8. Upgrading Packages
9. Installing to the User Site
10. Installing Requirements Files
11. Installing from VCS
12. Installing from other indexes
13. Installing from A Local Src Tree
14. Installing from a Local Archives
15. Installing from Other sources

16. Installing Prereleases

17. Installing Setuptools “Extras”

Some sites offer in-browser coding for those who want to learn Python:

- Codecademy
- Coding Bootcamps
- DataCamp
- Dataquest
- HackInScience

- High School Technology Services

Google Colab and Azure Note Book

Google CoLab and Azure Notebooks provide a flexible environment for developers to work.

They are very close to each other in terms of characteristics and can often be tricky to pick one.

Comparison between the two can be done on the basis of:

- Speed
- Computer Power
- Memory
- Importing Libraries
- Language Support
- File I/O

- Similarity with Jupyter
- Additional Features

Google Colab	Azure Note book
Supports python (currently 3.6.7 and 2.7.15).	Supports multiple kernels: python, R, F#
Allocates RAM of 12 GB and disk of memory 50 GB	Allocates small RAM of only 4GB.
GPU is available by default. You can use it for anything except for crypto mining and long-term usage!	GPU is not available to use. You should deploy your own VM to have GPU.
Codes will be saved in google drive and can edited collaboratively (by sharing). It is integrated with cloud storage such as Amazon s3.	Retrieving s3 data takes so much time
It has capability to use local hardware	It is not sure how to use local hardware
Integrated with code development in a team such as github	Integrated with github

Performing Operations using Data types and Operators

Questions Discussion

Why do we need to classify different data items?

What are different data types in Python?

What are different classes inside these different data types?

Overview of Data Types

Data Types

- Data types are classification of data elements
- Data types represent the type of value that specifies the operations that can be performed on a given set of data
- Data types are classes and variables are instance of these classes

Note: In Python, you need not declare the data type. The compiler automatically recognises the data type during the variable declaration.

Standard Data Types

Data Types

- Numeric
  - Integer

- Float

- Complex Number

- Sequence
    - Strings

    - Tuple

    - List

- Boolean

- Set

- Dictionary

Control Flow with Decisions and Loops

Questions Discussion

What are branching statements?

What are conditional statements?

Branching Statements

Branching statements or jump statements change the normal flow of execution based on some condition.

- Are primarily used to interrupt loop instantly
- Are used to unconditionally transfer program control from one point to elsewhere in the program

Types of Branching Statements

Python provides the following branching statements:

- Break - To break the loop and transfer control to the line which is immediately outside of the loop.
- Continue - To escape the current execution and transfer the control back to the start of the loop.
- Return - To explicitly end the execution and return the result.

All the branching statements have two forms:

- Labeled

- Unlabeled

Break statement

Break statement is used to:

- Terminate a loop
- Used within the for and while loops to alter the normal behavior of the code.

A break statement ends the loop it is in and the control flows to the next statement immediately below the loop.

Continue statement

Continue statement is used to:

- End the current iteration in a for loop or a while loop and move to the next iteration.
- Skip the rest of the code inside a loop.

The loop does not terminate but moves to the next iteration.

Return statement

Return statement is used to:

- Inside a function to return a value.
- To exit the function.

If return value not explicitly mentioned, then the value “None “ is returned automatically.

Pass statement

Pass statement is used to:

- Is a branching statement
- Is a null statement
- Is used as a placeholder
- Can be implemented with a blank body for a function or empty class.

A function without the pass statement and empty code will throw an IndentationError exception.

Conditional statement

Conditional statement help to determine whether the statement must be executed or not.

- Types of conditional statement
  - "If" statement
    - Has a Boolean expression followed by one or more statements.
  - "if else statement"
    - Executes when the Boolean expression for "if statement" is false.
  - "if elif else statement (nested if statement)"
    - Uses one if elif else statements as nested "if" statements.

Looping statement

Looping statements repeatedly execute a block of statements until a given condition within the loop is satisfied.

- Types of looping statements
  - "While
    - Used to execute a block of statements repeatedly until a given condition in a while loop is satisfied i.e., true.

  - "While with else
    - Used to execute the else part of a while loop, if the condition in the while loop yields the value, False.

- For
  - Used for iterating over a sequence either:
    - A list,
    - A tuple,
    - A dictionary,
    - A set, or
    - A string.

Performing Input and Output Operations

Questions Discussion

- What are the code segments that perform file input and output operations?
- What are the code segments that perform console input and output operations?

Importance of Input Operation

- Input plays an essential role because it allows you to interact and add new information.
- Python provides some built-in functions to read and modify the file, and to perform input-output operations.

File Handling

- File handling in Python is extremely important
- Easy to implement Python file handling

- Python input/output operations involves reading and writing process and many other file handling options
- How to read the data from a given file whenever you want to analyze the data

Built-in Python methods can manage two file:

  - Text
  - Binary

  Encode these two files differently

Each line of code includes:

  - A sequence of characters
  - Form a text file

Each line of a file is terminated with:

  - Special character, called the EOL or End of Line characters like comma {,} or newline character

- Ends the current line
- Tells the interpreter a new one has begun

Functions of File Handling

- Opening File
- Reading from Files
- Writing to Files
- Appending to Files
- Splitting Files

Input Operation

- Is simplest way to take input from the user, then evaluates the expression and finally converts the entered value into the string format (irrespective of format or type of entered value)
- Accepts a string message which is optional and meant to be displayed on the output console to ask a user to enter the input value
- Python programming language provides an input() function to take input into a program from the console.

Use of split() Method

- Take multiple values in one line
- Breaks the given input by the specified separator
- If the separator is not provided, then any white space is a separator.

Output Operation

- Is the simplest way to present or display a program output to the console, where you can pass zero or more expressions separated by commas
- Space ' ' as separator in print() Function
- Dash '-' as Separator in print() Function
- 'end' Parameter in print() Function
- 'file' Parameter in print() Function

Formatting Output Using String Modulo Operator

- The String modulo operator (%) can be used for string formatting by adding a placeholder in the string which is later replaced with the values in the tuple.
- String modulo operator (%) can be used to replace any integer, float, or string values.

Formatting Output Using Format Method

- Use {} to mark where a variable will be substituted and can provide detailed formatting directives.
- We can either use the exact position of the variable to be substituted or empty {} will substitute the variables in the order mentioned.

Document and Structure Code

Questions Discussion

- What are the code segments that use comments and documentation strings?

- What are the code segments that include function definitions?

Comments:

Comments can be used to:

- Explain Python code
- Make the code more readable
- Prevent execution when testing code

Docstrings:

- Python documentation strings or docstrings is a string used to explain the declared Python modules, functions, classes, and methods.
- To help the programmers working on the code to understand the details of its implementation.

Declaring and Accessing Docstrings

Two ways of declaring Docstrings:

- ”’triple single quotes”’
- “””triple double quotes”””

Accessing Docstrings:

- Use the __doc__ method
- of the object or using the help function.

Indentation in Docstrings:

- The entire docstring is indented the same as the quotes at its first line.
- Docstring processing tools will strip a uniform amount of indentation from the second and further lines of the docstring, equal to the minimum indentation of all non-blank lines after the first line.
- Any indentation in the first line of the docstring (i.e., up to the first newline) is insignificant and removed.
- Relative indentation of later lines in the docstring is retained.

Comments vs Docstrings

Comment	Docstrings
Provides useful information to help the reader understand the source code Explains logic or a part of it used in the code	• Provides a convenient way of associating documentation with Python modules, functions, classes, and methods.
Explains logic or a part of it used in the code.	
Uses # symbol	

Need for a Function

- Executes only when it called
- Uses parameters as data
- Returns data as a result
- In Python a function is defined using the def keyword.

Calling Function

To call a function, use the function name followed by parenthesis

Parameters or Function

A parameter is the variable listed inside the parentheses in the function definition.

An argument is the value that is sent to the function when it is called.

Arbitrary Argument

If the number of arguments is unknown, add a * before the parameter name in the function definition.

Recursion

- Recursion is a common mathematical and programming concept. It means that a function calls itself.
- The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amounts of memory or processor power.
- When written correctly recursion can be a very efficient and mathematically-elegant approach to programming.

Performing Troubleshooting and Error Handling

Questions Discussion

- What are the different errors in code segments?
- What are the code segments that handle exceptions?

Performing Troubleshooting and Error Handling – Types of Errors

Python documentation has various errors such as:

- Name Error
    - Encountered when Python does not recognise the name of something in a statement
    - Can happen when we try and refer to a variable that has not been defined
- Syntax Error
    - Encountered with Python when it is unable to parse a section of the code
    - Construction- Does not conform to a format that Python can interpret
- Indentation Error
    - Indentation is used to indicate program structure.
    - It needs to be used consistently throughout in the code.
- Type Error
    - Encountered when something is wrong with the data type used for a particular operation or function
    - Try to combine data types using an operator that it doesn't know what to do with
- Index Error
    - Encountered when we try and access an element of a collection beyond the number of elements that the collection contains.
- Attribute Error
    - Encountered when trying to use a function or variable attached a data that is not present.

Syntax Errors verses Exceptions

Syntax Errors	*Exceptions*
• Syntax errors occur when the parser detects an incorrect statement. • The arrow indicates where the parser ran into the syntax error. • In this example, there was one bracket too many. Remove it and run the code again.	• This time, the code ran into an exception error. • This type of error occurs whenever syntactically correct Python code results in an error. • The last line of the message is indicated what type of exception error it is. • Python details what type of exception error was encountered. • It is a ZeroDivisionError

Handling Exceptions

- The try and except blocks in Python are used to catch and handle exceptions.
- Python executes code following the try statement as a “normal” part of the program.
- The code that follows the except statement is the program’s response to any exceptions in the preceding try clause.
- When syntactically correct code runs into an error, Python will throw an exception error.
- This exception error will crash the program if it is unhandled.
- We can also start by **asserting** Python.
- If this condition turns out to be true, then that is excellent!
- If the condition turns out to be False, you can have the program throw an AssertionError exception.

The else Clause

- You can instruct a program to execute a particular block of code only in the absence of exceptions.
- If you were to run this code on a Linux system, the output would be on the screen.
- You can also try to run code inside the else clause and catch possible exceptions there as well.
- If you were to execute this code on a Linux machine, you would get the result given on the screen.
- You can see that the linux_interaction() function ran. Because no exceptions were encountered, an attempt to open file.log was made.
- That file did not exist, and instead of opening the file, you caught the FileNotFoundError exception.

Performing Operations using Modules and Tools

Questions Discussion

- What are the basic operations that can be performed using built-in modules?
- How are complex computing problems solved using built-in modules?

Built-in Modules

- Long and complex logic in a program is broken into smaller, independent, and reusable blocks of instructions. These are called modules.
- Modules are designed to perform a specific task that is a part of the entire process.
- Each built-in module contains resources for certain specific functionalities such as:
    - OS Management
    - Disk IO
    - Networking
    - Database Connectivity
- Built-in modules are mostly written in C and integrated with a Python interpreter.
- Built-in modules where the functions are frequently used:
- Most common modules in python are:
    - OS modules
    - Random module
    - Math module
    - Time module
    - Sys module
    - Collections module

- Statistics module

Solving Complex Computing Problems

- This module provides access to mathematical functions for complex numbers.
- The functions in this module accept
- Integers, floating-point numbers, or complex numbers as arguments.
- Any Python object that has either a complex() or a float() method.
- These methods are used to convert the object to complex or floating-point numbers.
- The function is then applied to the result of the conversion.
- Python can handle complex numbers and their associated functions using the file “cmath”.

- Complex numbers are useful in many mathematical applications, and Python provides useful tools for handling and manipulating them.
    - Converting real numbers to complex numbers
    - Phase of complex number
    - Converting from polar to rectangular form and vice versa