Python (Part-1) Assignments by Topic, by K. Praveen Kumar : (Date: 22 Sep2025)

Each assignment should be submitted as .py file with comments or in jupyter notebook with explaining what the code does.

2. Basic Syntax

A. Run Python in **interactive mode** and calculate factorial of 5 without using loops/functions.

B. Write a script that takes user input of two numbers and prints their sum (script mode).

C. Demonstrate difference between **identifier naming rules** (valid vs invalid).

D. Write a program using **multi-line statements** (\) to calculate the average of 5 numbers.

E. Print a multiline string using **triple quotes**.

F. Write code showing **different comment styles**.

G. Take user input for a sentence and print it in uppercase.

H. Write a script that accepts **command-line arguments** and prints the sum of integers passed.

---

3. Variable Types

A. Create variables of different data types (int, float, str, bool). Print type using type().

B. Demonstrate **multiple assignment** (a=b=c=10) and **unpacking** (x,y,z=1,2,3).

C. Swap values of two variables without using a third variable.

---

4. Python Tools Installation

A. Install Anaconda and run a Python program using **Spyder IDE**.

B. Install PyCharm and configure a Python interpreter. Run your earlier script in PyCharm.

C. On Linux: install Python 3, check version, and run a script using python3 file.py.

---

5. Standard Data Types

Numbers

A. Perform operations with int, float, complex. Show results.

B. Convert a float to int, int to complex, etc. Print type changes.

C. Write a program to generate **5 random numbers between 10–50**.

D. Write a program to calculate sine, cosine, and tan of 30, 45, 60 degrees.

E. Use math.pi and math.e to calculate area of a circle with radius=7.

Strings

A. Accept a string and print its reverse.

B. Replace "Python" with "Java" in "I like Python".

C. Write a program using escape sequences (\n, \t).

D. Demonstrate **string concatenation and repetition**.

E. Format a string with placeholders: "My name is {name}, I am {age} years old.".

F. Write a program to count vowels in a given string.

G. Show 5 built-in string methods with examples.

Lists

A. Create a list of 10 integers, print even numbers only.

B. Write a program to add, update, and delete elements from a list.

C. Slice a list to get last 3 elements.

D. Convert a 3×3 matrix into a list of lists and access diagonal elements.

E. Use list methods (append, extend, insert, remove, pop, sort, reverse).

Tuples

A. Create a tuple and demonstrate accessing via indexing and slicing.

B. Show immutability by trying to update tuple elements.

C. Concatenate two tuples.

D. Convert a tuple to a list and modify elements.

E. Use built-in tuple methods (count, index).

Dictionary

A. Create a dictionary with student roll numbers and names. Print values using keys.

B. Update dictionary by adding new key-value pair.

C. Delete an element using pop.

D. Demonstrate dictionary keys must be immutable (try list as key → error).

E. Use built-in functions (keys(), values(), items()).

Sets

A. Create a set of unique numbers from a list with duplicates.

B. Perform union, intersection, difference of two sets.

C. Add and remove items from a set.

D. Demonstrate frozen set.

---

6. Operators

A. Demonstrate each operator type with examples.

B. Show difference between is and ==.

C. Write program to check if a number is in a list using **membership operator**.

D. Evaluate 2 + 3 * 4 ** 2 and explain precedence.

---

7. Decision Making

A. Write program to check if a number is positive, negative, or zero.

B. Check if a year is leap year.

C. Nested if: find the greatest of 3 numbers.

D. Implement a grading system (marks → grade).

---

8. Loops

A. Print multiplication table of a number using for.

B. Print Fibonacci sequence using while.

C. Use nested loops to print a pattern (triangle of *).

D. Demonstrate break, continue, pass.

E. Write generator for infinite odd numbers.

---

9. Functions

A. Write function to find factorial.

B. Demonstrate keyword, default, variable-length arguments.

C. Use lambda to square numbers in a list.

D. Write recursive function for Fibonacci.

E. Show difference between local and global variables.

---

10. OOP

A. Define a class Student with attributes name, roll. Create objects.

B. Add methods for displaying details.

C. Use __init__ constructor to initialize.

D. Show private (__var) and protected (_var) variables.

E. Differentiate between class variable and instance variable.

---

11. Inheritance

A. Create base class Animal and derived class Dog.

B. Show multiple inheritance with Teacher and Researcher.

C. Overload + operator to add two objects.

---

12. collections.abc

A. Implement a custom container class and inherit from collections.abc.Iterable.

B. Demonstrate custom exception handling (user-defined exception for invalid input).

---

13. Functional Programming

A. Use itertools to generate permutations of [1,2,3].

B. Use functools.reduce to find product of list.

C. Use operator module to sort list of tuples by second element.

D. Write a decorator for logging function execution time.

E. Nested list comprehension: generate 2D matrix of squares.

F. Write generator for prime numbers.

---

14. Directories

A. Create directory mydata using mkdir().

B. Change to directory and get current path.

C. Delete directory.

D. Write program to list all .txt files in a directory.

---

15. Error & Exception Handling

A. Write program that raises ZeroDivisionError and handle it.

B. Handle multiple exceptions (ValueError, TypeError).

C. Use try-finally for file closing.

D. Write user-defined exception NegativeNumberError.

E. Demonstrate assert for input validation.