Lab Manual

Programming Using Python: CSPE-22

Programming Using Python: ITPE-22

Programming using Python
(CSPE-22)

Number of Credits: 4

Course Learning Objectives:
1. Building robust applications using Python programming language's features.
2. Understanding the usage of Python libraries.
3. Building multithreaded, platform-independent and GUI based python applications for business Problems.

Course Content:

1. The concept of data types; variables, assignments; immutable variables; numerical types; arithmetic operators and expressions; comments in the program; understanding error messages; Conditions, boolean logic, logical operators; ranges; Control statements: if-else, loops (for, while); short-circuit (lazy) evaluation; Strings and text files; manipulating files and directories, os and sys modules; text files: reading/writing text and numbers from/to a file; creating and reading a formatted file (csv or tab-separated); String manipulations: subscript operator, indexing, slicing a string.

2. Lists, tuples, and dictionaries; basic list operators, replacing, inserting, removing an element; searching and sorting lists; dictionary literals, adding and removing keys, accessing and replacing values; traversing dictionaries; Design with functions: hiding redundancy, complexity; arguments and return values; formal vs actual arguments, named arguments.

3. Simple Graphics and Image Processing: "turtle" module; simple 2d drawing - colors, shapes; digital images, image file formats, image processing: Simple image manipulations with 'image' module (convert to bw, greyscale, blur, etc). Classes and OOP: classes, objects, attributes and methods; defining classes; design with classes, data modeling; persistent storage of objects; inheritance, polymorphism, operator overloading (_eq_, _str_, etc); abstract classes; exception handling, try block

4. Graphical user interfaces; event-driven programming paradigm; tkinter module, creating simple GUI; buttons, labels, entry fields, dialogs; widget attributes - sizes, fonts, colors layouts, nested frames.

Reference Books:

1. T.R. Padmanabhan, Programming with Python, Springer,
2. 1st Ed., 2016.
2. Kenneth Lambert, Fundamentals of Python: First Programs, Cengage Learning,, 1st Ed., 2012.

Course outcomes:

1. Write python programs that solve simple business problems.
2. Create python applications that are robust and multithreaded.
3. Write simple GUI interfaces for a program to interact with users, and to understand the event-based
   GUI handling principles in python.

Programming using Python
(ITPE-22)

Number of Credits: 4

Course Learning Objectives:
1. Building robust applications using Python programming language's features.
2. Understanding the usage of Python libraries.
3. Building multithreaded, platform-independent and GUI based python applications for business Problems.

Course Content:
1. The concept of data types; variables, assignments; immutable variables; numerical types; arithmetic
   operators and expressions; comments in the program; understanding error messages; Conditions,
   boolean logic, logical operators; ranges; Control statements: if-else, loops (for, while); short-circuit (lazy) evaluation; Strings and text files; manipulating files and directories, os and sys modules; text files: reading/writing text and numbers from/to a file; creating and reading a formatted file (csv or tab-separated); String manipulations: subscript operator, indexing, slicing a string.

2. Lists, tuples, and dictionaries; basic list operators, replacing, inserting, removing an element;
   searching and sorting lists; dictionary literals, adding and removing keys, accessing and replacing values; traversing dictionaries; Design with functions: hiding redundancy, complexity; arguments and return values; formal vs actual arguments, named arguments.

3. Simple Graphics and Image Processing: "turtle" module; simple 2d drawing - colors, shapes; digital
   images, image file formats, image processing: Simple image manipulations with 'image' module
   (convert to bw, greyscale, blur, etc). Classes and OOP: classes, objects, attributes and methods; defining classes; design with classes, data modeling; persistent storage of objects; inheritance, polymorphism, operator overloading (_eq_, _str_, etc); abstract classes; exception handling, try block

4. Graphical user interfaces; event-driven programming paradigm; tkinter module, creating simple GUI;

buttons, labels, entry fields, dialogs; widget attributes - sizes, fonts, colors layouts, nested frames.

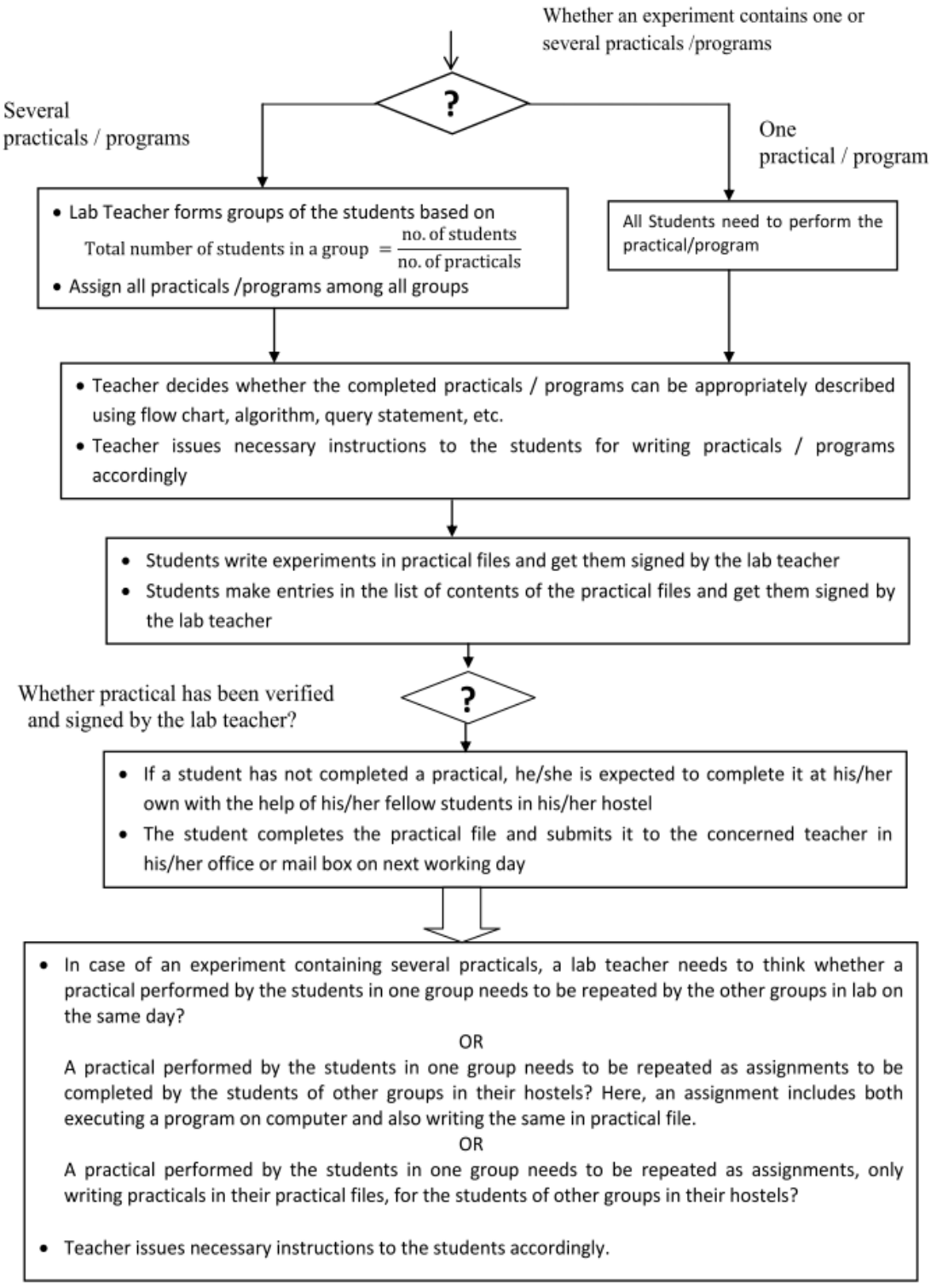
Reference Books:

1. T.R. Padmanabhan, Programming with Python, Springer, 1st Ed., 2016.
2. Kenneth Lambert, Fundamentals of Python: First Programs, Cengage Learning,, 1st Ed., 2012.

Course outcomes:

1. Write python programs that solve simple business problems.
2. Create python applications that are robust and multithreaded.
3. Write simple GUI interfaces for a program to interact with users, and to understand the event-based
   GUI handling principles in python.

Lab Instructions



Lab Manual
Programming Using Python
CSPE22 and ITPE22

Experiment-1

I. Write a program that reads temperature in degree Celsius from console and converts it to Fahrenheit. Display the result. Formula for conversion is given as follows:
*Fahrenheit = (9/5) * Celsius + 32*

II. Write a program that reads the radius and length of a cylinder, to compute the area and volume of the cylinder using following formula:
*Area = radius * radius * π*
*Volume = area * length*

III. Write a program that reads a number in feet, converts it to meters, and displays the result. One foot is 0.305 meters.

IV. Write a program that reads a weight in pounds and converts it into kilogram. One pound is 0.454 kilograms. Display the results.

V. Write a program to find the sum of all the prime numbers less of equal to a given number.

VI. Write a program that calculates the distance between two points, where the coordinates of the points are provided by the user.

VII. Write a program that calculates the energy needed to heat the water from an initial temperature to a final temperature. Program prompts the user to provide the amount of waters in kilograms and the initial and final temperatures of water. Formula for calculating energy is as below:

*Energy (in joules) = weight of water (k.g.) * (finalTemp - initialTemp) * 4184*

Experiment-2

I. Write a program that prompts the user to enter the length, from the center of Pentagon to a vertex and computes the area of the Pentagon.

The formula for computing the area of a Pentagon is:
Area = (3*root(3)*s^2)/2
where s is the length of a side. The side can be computed using the formula:
s = 2*r*sin(3.14/5), where r is the length from the center of a Pentagon to a vertex.

II. The area of a Pentagon can be computed using the following formula(s is the length of a side)

Area = (5*s^2)/(4*tan(3.14/5))
Write a program that prompts the user to enter the side of a Pentagon and displays the area.

III. (Reverse number) Write a program that prompts the user to enter a four-digit integer and display the number in reverse order.

IV. (Turtle: draw the Olympic symbol) Write a program that prompts the user to enter the radius of the rings and draws an Olymoic symbol of five rings of the same size with the colors blue, black, red, yellow, and green.

Experiment-3

I. (Algebra: solve quadratic equations) The two roots of a quadratic equation, for example, $ax^2 + bx + c = 0$, can be obtained using the following formula:

$r_1 = (-b + \sqrt{b^2 - 4ac})/2a$ and $r_2 = (-b - \sqrt{b^2 - 4ac})/2a$

b^2-4ac is called the discriminant of the quadratic equation. If it is positive, the equation has two real roots. If it is zero, the equation has one root. If it is negative, the equation has no real roots.

Write a program that prompts the user to enter values for a,b, and c and displays the result based on the discriminant. If the discriminant is positive, display two roots. If discriminant is 0, display one root. Otherwise display the equation has no real roots.

II. (Algebra: solve 2*2 linear equations) You can use Cramer's Rule to solve the following 2*2 system of linear equation:

$$ax + by = e \qquad x = (ed - bf)/(ad - bc)$$
$$cx + dy = f \qquad y = (af - ec)/(ad - bc)$$

Write a program that prompts the user to enter a, b, c, d, e, and f and display the result. If ad - bc is 0, report that the equation has no solution.

III. Write a program that generates two integers under 100 and prompts the user to enter the sum of these two integers. The program then report true if the answer is correct, false otherwise.

IV. Write a program that prompts the user to enter an integer for today's day of the week (Sunday is 0, Monday is 1…, and Saturday is 6). Also prompt the user to enter the number of days after today for a future day and display the future day of the week.

V. Suppose you shop for rice and find it i two different sized packets. You would like to write a program to compare the costs of the packages. The program prompts the user to enter the weight and price of each package and the displays the one with the better price.

VI. (Find the number of days in a month) Write a program that prompts the user to enter the month and year and displays the number of days in the month. For example, if the user entered month 2 and year 2000, the program should display that February 2000 has 29 days. If the user entered 3 and year 2005, the program should display that March 2005 has 31 days.

VII. Write a program that prompts the user to enter an integer and check whether the number is divisible by both 5 and 6, divisible by 5 or 6 or just one of them (but not both of them).

VIII. Write a program that reads three edges for a triangle and computes the perimeter if the input is valid. Otherwise, display that the input is invalid. The input is valid if the sum of every pair of two edges is greater than the remaining edge.

IX. Write a program that prompts the user to enter a hex character and displays its corresponding decimal integer.

Experiment-4

I. (Conversion from miles to kilometer) Write a program that displays the following table(note that 1 mile is 1.609 kilometers):

Miles	Kilometers
1	1.609
2	3.218
. . .	
9	15.481
10	16.090

II. Suppose that the tuition for a university is $10000 this year and and increases 5% every year. Write a program that computes the tuition in ten years and the total cost of four years' worth of tuition starting ten years from now.

III. Write a program that displays ten numbers per line, all the numbers from 100 to 1,000 that are divisible by 5 and 6. The numbers are separated by exactly one space.

IV. Write a program that displays the characters in the ASCII character table from! to ~. Display ten characters per line. The characters are separated by exactly one space.

V. Use nested loop that display the following patterns in four separated programs:

Pattern A	Pattern B	Pattern C	Pattern D
1	1 2 3 4 5 6	1	1 2 3 4
5 6			
1 2	1 2 3 4 5	2 1	1 2 3 4
5			
1 2 3	1 2 3 4	3 2 1	1 2 3 4
1 2 3 4	1 2 3	4 3 2 1	1 2 3
1 2 3 4 5	1 2	5 4 3 2 1	1 2
1 2 3 4 5 6	1	6 5 4 3 2 1	1

VI. Write a nested **for** loop that displays the following output:

```
                1
              1   2   1
            1   2   4   2   1
          1   2   4   8   4   2   1
        1   2   4   8   16  8   4   2   1
      1   2   4   8   16  32  16  8   4   2   1
```

VIII. Write a program that lets the user enter the loan amount and loan period in number of years and displays the monthly and total payments for each interest rate starting from 5% to 8%, with and increment of 1/8.

IX. The monthly payment for the given loan pays the principal and the interest. The monthly interest is computed by multiplying the monthly interest rate and the balance (the remaining principal). The principal paid for the month is therefore the monthly payment minus the monthly interest. Write a program that lets the user enter the loan amount, number of years, and interest rate, and then displays the amortization schedule for the loan.

Note:- The balance after the last payment may not be zero. If so, the last payment should be the normal monthly payment plus the final balance.

Hint:-

Write a loop to display the table. Since the monthly payment is the same for each month, it should be computed before the loop. The balance is initially the oan amount. For each iteration in the loop, compute the interest and principal and update the balance. The loop may look like this:

```
for i in range(1,  numberof years*12+1)
interest  =  monthlyInterestRate*balance
principal = monthlyPayment - interest
balance = balance - principal
print(i, "\t\t", interest, "\t\t",  principal,  "\t\t",balance)
```

X. A positive integer is called a perfect number if it is equal to the sum of all of its positive divisors, excluding itself. For example, 6 is the first perfect number, because 6 = 3 + 2 + 1 . The next is 28 = 14 + 7 + 4 + 2 + 1. There are four perfect numbers less than 10, 000. Write a program to find these four numbers.

XI. Write a program that displays all possible combinations for picking two numbers from integers 1 to 7. Also display the total number of combinations.

1 2
1 3
. . .
. . .
The total number of all combinations is 21.

Experiment-5

I. Write a Python program to convert a string in a list.

II. Write a Python program to find the first repeated word in a given string.

III. Write a Python program to find all the common characters in lexicographical order from two given lower case strings. If there are no common letters print "No common characters".

IV. Write a Python program to create two strings from a given string. Create the first string using those characters which occurs only once and create the second string which consists of multi-time occurring characters in the said string.

V. Write a Python program to find the index of a given string at which a given substring starts. If the substring is not found in the given string return 'Not found'.

Experiment-6

I. Write a Python program to read an entire text file.

II. Write a Python program to read a file line by line and store it into a list.

III. Write a Python program to combine each line from first file with the corresponding line in second file.

IV. Write a Python program that takes a text file as input and returns the number of words of a given text file.
Note: Some words can be separated by a comma with no space.

V. Write a Python program to generate 26 text files named A.txt, B.txt, and so on up to Z.txt.

VI. Write a Python program to create a file where all letters of English alphabet are listed by specified number of letters on each line.

Experiment-7

I. Write a Python program to multiplies all the items in a list.

II. Write a Python program to count the number of strings where the string length is 2 or more and the first and last character are same from a given list of strings.
Sample List : ['abc', 'xyz', 'aba', '1221']
Expected Result : 2

III. Write a Python function that takes two lists and returns True if they have at least one common member.

IV. Write a Python program to generate and print a list of first and last 5 elements where the values are square of numbers between 1 and 30 (both included).

V. Write a python program to check whether two lists are circularly identical.

VI. Write a Python program to convert a pair of values into a sorted unique array.

VII. Write a Python program to convert list to list of dictionaries.
Sample lists: ["Black", "Red", "Maroon", "Yellow"], ["#000000", "#FF0000", "#800000", "#FFFF00"]
Expected Output: [{'color_name': 'Black', 'color_code': '#000000'}, {'color_name': 'Red', 'color_code': '#FF0000'}, {'color_name': 'Maroon', 'color_code': '#800000'}, {'color_name': 'Yellow', 'color_code': '#FFFF00'}]

VIII. Write a Python program to find a tuple, the smallest second index value from a list of tuples.

IX. Write a Python program to check if all dictionaries in a list are empty or not.
Sample list : [{},{},{}]

Return value : True
Sample list : [{1,2},{},{}]
Return value : False

Experiment-8

I. Write a Python program to find the index of an item of a tuple.

II. Write a Python program to unzip a list of tuples into individual lists.

III. Write a Python program to remove an empty tuple(s) from a list of tuples.
Sample data: [(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]
Expected output: [('',), ('a', 'b'), ('a', 'b', 'c'), 'd']

IV. Write a Python script to check if a given key already exists in a dictionary.

V. Write a Python script to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).
Sample Dictionary (n = 5) :
Expected Output : {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

VI. Write a Python program to get the maximum and minimum value in a dictionary.

VII. Write a Python program to count the values associated with key in a dictionary. Sample data: = [{'id': 1, 'success': True, 'name': 'Lary'}, {'id': 2, 'success': False, 'name': 'Rabi'}, {'id': 3, 'success': True, 'name': 'Alex'}]
Expected result: Count of how many dictionaries have success as True

VIII. Write a Python program to create a dictionary from two lists without losing duplicate values.
Sample lists: ['Class-V', 'Class-VI', 'Class-VII', 'Class-VIII'], [1, 2, 2, 3]
Expected Output: defaultdict(<class 'set'>, {'Class-VII': {2}, 'Class-VI': {2}, 'Class-VIII': {3}, 'Class-V': {1}})

Experiment-9

I. Write a Python function that accepts a string and calculate the number of upper case letters and lower case letters.

II. Write a Python function that checks whether a passed string is palindrome or not.
Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam or nurses run.

III. Write a Python function to check whether a string is a pangram or not.
Note : Pangrams are words or sentences containing every letter of the alphabet at least once.
For example : "The quick brown fox jumps over the lazy dog"

IV. Write a Python program to detect the number of local variables declared in a function.

Experiment-10

I. Write a program to calculate the distance between two points and represent these two pints along with the mid points in the Cartesian plane.

II. Write a program to represent the five circles having different radius and same center.

III. Write a program to a pentagon and put a line from each vertex to the center. Fill different color in each block.

IV. Write a program to read a gray image and convert it to the binary image. Write both the image.

V. Write a program to read a color image and extract its pixel value in the red, green and blue dimension.

Experiment-11

I. Write a Python class to convert a roman numeral to an integer.

II. Write a Python class to find validity of a string of parentheses, '(', ')', '{', '}', '[' and ']. These brackets must be close in the correct order, for example "()" and "()[]{}" are valid but "[)", "({[)]" and "{{{" are invalid.

III. Write a Python class which has two methods get_String and print_String. get_String accept a string from the user and print_String print the string in upper case.

IV. Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.

V. Design a class named triangle that extends the GeometricObject class. The Tringle class contains:

- Three float fields named side1, side2, and side3 to denote the three sides of the triangle.
- A constructor that creates a tringle with the specified side1, side2, and side3, with default value 1.0.
- The accessor methods for all three data fields.
- A method named getArea() that returns the perimeter of this triangle.
- A method named _ _ str_ _ () that returns a string description for the triangle.

Experiment-12

I. Write a Python GUI program to import Tkinter package and create a window and set its title.

II. Write a Python GUI program to import Tkinter package and create a window. Set its title and add a label to the window.

III. Write a Python GUI program to create a label and change the label font style (font name, bold, size) using tkinter module.

IV. Write a Python GUI program to create a window and set the default window size using tkinter module.

V. Write a Python GUI program to create a window and disable to resize the window using tkinter module.