MCA 1st Semester

CS7012: Functional Programming Paradigm

Practical Workbook

Course Objective: To develop a system using functional programming concepts to build side-effect-free functions for easily maintain system.

Course Outcomes:

Upon completion of the course student shall be able to:
CO1: Understand the basic features of functional programming.
CO2: Analyze and design complex concepts by represented using different types of functions by combining simple functions.
CO3: Able to apply appropriate Data Structures for processing data efficiently.
CO4: Create a module to organize the logic of the variable, functions and classes.
CO5: Apply and design callables for Object oriented programme as well as apply the Lazy evaluation.
CO6: Design and implement the complex problem having multiple branches using Recursive functions.

Programme Outcomes:

PO1: Proficiency in and ability to identify problems related to computer science as well as design and apply computational knowledge to solve them.
PO2: Ability to design, develop, test and maintain system, component, product or process as per needs and specification.
PO3: Understanding of professional and ethical role and responsibility.
PO4: Recognition of the need for and ability towards life-long learning.
PO5: Knowledge of programming languages, database systems, operating systems, software engineering, Web & Mobile technology and relevant modern issues.
PO6: Ability to demonstrate the use of modern tools, models and languages to solve problems related to software development.
PO7: Ability to communicate and present knowledge effectively.

Semester Skills:

SO1: Thinking Skill: Critical
SO2: Learning Skill: Solitary(Intrapersonal)
SO3: Listening Skill: Evaluative
SO4: Writing Skill: Expository
SO5: Courses' based technical skills

Number of Practical Problems in Journal

Unit	No. of Practical Problems	Time required to implement (in hours)	Minimum required for journal evaluation
1	02	06	02
2	02	08	02
3	02	08	02
4	02	08	02
5	02	08	02
6	02	08	02
Total:	**12**	**46**	**12**

Tools and Technologies:

- Python 3.8 and above
- Any Python IDLE

Suggested Learning Resources:

1. Steven F. Lott, Functional Python Programming Second Edition, Packt
2. David Mertz, Functional Programming in Python, O'Reily
3. https://docs.python.org/3/howto/functional.html

Practical No.: 1		**Enrolment No.**		
Objective:	To understand the basics of Python programming			
Practical Exercise:	Ask user to input any random number. And display the message using control and looping construct that: If the inputted number is same as randomly generated number than display "You are right! <random number>". If the inputted number is less than randomly generated number then display "Ohh! Your guess is too low." else display "Your guess is too high!" <u>Intermediate Level Exercise:</u> 1. Modify above program in such a way that it gives the user only 3 chances to guess the correct number. If they try three times without success, they're told that they didn't guess in time, and the program exits. 2. Provide a choice to user for different number system such as, If user inputs "10" as their guess, you'll need to interpret it in the correct number base; "10" might mean 10 (decimal) or 2 (binary) or 16 (hexadecimal). Not only should you choose a random number, but you should also choose a random number base, from 2 to 16, in which the user should submit their input.			
Pre-requisite:	Basic knowledge of control and looping construct in Python.			
Duration for Completion:	**03**			
Reference to solve the problem:	https://docs.python.org/3/tutorial/			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: 1. Proper usage of control and looping construct, 2. Business Logic 3. Intermediate exercise 4. Technical via				
Marks:	**3**	**3**	**2**	**2**
Secured by Student:				
Teacher's Signature:				
Date:				

Practical No.: 2		**Enrolment No.**
Objective:	To understand python basics and its data structure.	
Practical Exercise:	Do as directed for the given problem: 1. Write a Python program to convert more than one list to nested dictionary. Original strings: ['S001', 'S002', 'S003', 'S004'] ['Adina Park', 'Leyton Marsh', 'Duncan Boyle', 'Saim Richards'] [85, 98, 89, 92] Nested dictionary: Expected Output: [{'S001': {'Adina Park': 85}}, {'S002': {'Leyton Marsh': 98}}, {'S003': {'Duncan Boyle': 89}}, {'S004': {'Saim Richards': 92}}] 2. Write a Python program to generate a Collatz conjecture sequence, where you take any positive integer n, if n is even then divide it by 2 and get n / 2. If n is odd then multiply it by 3 and add 1 to obtain 3n + 1. Repeat the process until you reach 1. Expected Output: n = 24 the sequence is: 24, 12, 6, 3, 10, 5, 16, 8, 4, 2, 1. n = 37 the sequence is: 37, 112, 56, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1 3. Write a Python program to create a dictionary grouping a sequence of key-value pairs into a dictionary of lists. Original list: [('yellow', 1), ('blue', 2), ('yellow', 3), ('blue', 4), ('red', 1)] Grouping a sequence of key-value pairs into a dictionary of lists: Expected Output: {'yellow': [1, 3], 'blue': [2, 4], 'red': [1]} 4. 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'}] 5. Write a Python program to find the items starts with specific character from a given list. Expected Output: Original list: ['abcd', 'abc', 'bcd', 'bkie', 'cder', 'cdsw', 'sdfsd', 'dagfa', 'acjd'] Items start with a from the said list: ['abcd', 'abc', 'acjd'] Items start with d from the said list: ['dagfa'] Items start with w from the said list: []	

	6. Write a Python program to calculate the average value of the numbers in a given tuple of tuples. Original Tuple: ((10, 10, 10, 12), (30, 45, 56, 45), (81, 80, 39, 32), (1, 2, 3, 4)) Average value of the numbers of the said tuple of tuples: [30.5, 34.25, 27.0, 23.25] Original Tuple: ((1, 1, -5), (30, -15, 56), (81, -60, -39), (-10, 2, 3)) Average value of the numbers of the said tuple of tuples: [25.5, -18.0, 3.75]		
Pre-requisite:	Fundamental knowledge of Python data structure.		
Duration for Completion:	**03**		
Reference to solve the problem:	https://docs.python.org/3/tutorial/		
Solution must contain:	Written description for the given problem and printout of output screen.		
Evaluation Parameters: 1. Business logic 2. Desired output 3. Technical viva			
Marks:	**4**	**3**	**3**
Secured by Student:			
Teacher's Signature:			
Date:			

Practical No.: 3		**Enrolment No.**		
Objective:	To implement mutable and immutable data structure			
Practical Exercise:	1. Assume that there is a one dictionary having fifty words with its short but relevant description. Each word may be of maximum four letters. Now program choose a random word from the dictionary and ask user to guess the word based on its shown description. If user's answer matched with randomly chosen word then display "Smart guess!" else give user one more chance for guessing but this time offer new word. 2. Build a program to work as online quiz of five questions. Each question have four possible options such as A, B, C and D. One of the four is correct answer and other three are wrong answers. After completing the quiz user get output as the total marks scored, and if any of the answer is wrong then it will display the correct answer for those questions. <u>Intermediate Level Exercise:</u> Modify case-2 and provide more than four options for single question, for that ask user on the time of question entry. [Ask user for question entry] Now each answer might have different weight for its correctness level such as option A - 20%, B-20%, E-60% means options C and D have 0% weight consider that those are wrong answers.			
Pre-requisite:	Python data structure			
Duration for Completion:	**04**			
Reference to solve the problem:	https://docs.python.org/3/tutorial/			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Usage of appropriate data structure 2. Business Logic with desired output 3. Intermediate exercise 4. Technical viva*				
Marks:	**3**	**3**	**3**	**1**
Secured by Student:				
Teacher's Signature:				
Date:				

Practical No.: 4		**Enrolment No.**	
Objective:	To understand the concept of functions and its utilization		
Practical Exercise:	Assume that a bank maintain two kinds of accounts for customers, one called as savings account and other as current account. The saving account provides deposit and withdrawal facilities, and provides compound interest on the deposited amount. The current account provides cheque book facility to withdraw amount. Current account holders should also maintain a minimum balance and if the balance falls below this level, bank gives alert for the same to customer. Develop an application in order to achieve the following tasks: a. Bank employee can open an account (Savings/Current) by filling customer information such as full name, date of birth, gender, address, contact no, email id, PAN no etc. The unique account number must be auto-generated which must be of eight digits. b. At the time of money deposit system should ask account no and type of account, minimum amount to deposit should be Rs. 1000. Date and time of money deposit transaction must be noted. c. At the time of money withdrawal system should ask account no and type of account. On a day each account holder is allowed to withdraw maximum of Rs. 20,000 only. At the time of withdrawal system should maintain minimum Rs. 2000 in account and accordingly allow for the transaction otherwise provide appropriate message to the user. d. At the time of money deposit and withdrawal, available amount shall be visible and date time shall be noted during the transaction. d. Bank employee can see all the customer details with their amount. The report shall also be available which should display daily transaction.		
Pre-requisite:	Modules and functions		
Duration for Completion:	**04**		
Reference to solve the problem:	Functional Python Programming by Steven F. Lott		
Solution must contain:	Written description for the given problem and printout of output screen.		
Evaluation Parameters: *1. Usage of functions 2. Desired output 3. Technical viva*			
Marks:	**05**	**03**	**02**
Secured by Student:			
Teacher's Signature:			
Date:			

Practical No.: 5		**Enrolment No.**		
Objective:	To implement the composition of functions.			
Practical Exercise:	1. Read "fp.txt" file and, first convert contents of file to upper case then perform the following actions on converted text. a. No. of characters b. No. of words c. No. of sentences d. Unique list of articles used e. No. of sentences which are connected with comma f. Unique list of number or special symbols 2. Develop a menu-based Scientific calculator by applying functional programming concepts, which is able to do at least ten basic mathematics operations.			
Pre-requisite:	Currying			
Duration for Completion:	**04**			
Reference to solve the problem:	Functional Python Programming by Steven F. Lott , https://www.python-course.eu/currying_in_python.php			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Adopted Functional programming style 2. Usage of currying concept 3. Complete solution with desired output 4. Technical viva*				
Marks:	**3**	**3**	**2**	**2**
Secured by Student:				
Teacher's Signature:				
Date:				

<table>
<tr><td>Practical No.: 6</td><td colspan="4"></td><td>Enrolment No.</td></tr>
<tr><td>Objective:</td><td colspan="5">To implement the concept of closures and decorators</td></tr>
<tr><td>Practical Exercise:</td><td colspan="5">1. Create a Python program to demonstrate the behaviour of decorator in given scenario.

Consider a plain pizza price is ₹200. If customer demands extra toppings on pizza that time it will cost extra according to given in below table.

<table>
<tr><th>Extra toppings</th><th>Extra cost in ₹</th></tr>
<tr><td>Extra cheese</td><td>100</td></tr>
<tr><td>Paneer</td><td>80</td></tr>
<tr><td>Baby corn</td><td>50</td></tr>
<tr><td>Pineapple</td><td>60</td></tr>
<tr><td>Sweet corn</td><td>30</td></tr>
</table>
Such as extra cheese cost add ₹100 In plain pizza price and so on.

<u>For reference:</u>

<pre>Plain Pizza

'Plain Pizza, Chicken added' using ChickenPizzaDecorator

'Plain Pizza, Vegetables added' using VegPizzaDecorator</pre></td></tr>
<tr><td>Pre-requisite:</td><td colspan="5">Nested functions with non-local variables</td></tr>
<tr><td>Duration for Completion:</td><td>04</td><td colspan="4"></td></tr>
<tr><td>Reference to solve the problem:</td><td colspan="5">https://docs.python.org/3/glossary.html</td></tr>
<tr><td>Solution must contain:</td><td colspan="5">Written description for the given problem and printout of output screen.</td></tr>
<tr><td colspan="6">Evaluation Parameters: 1. Decorators implementation 2. Calling of wrapper function 3. Handling multiple arguments 4. Technical viva.</td></tr>
<tr><td>Marks:</td><td>4</td><td>2</td><td>2</td><td>2</td></tr>
<tr><td>Secured by Student:</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Teacher's Signature:</td><td colspan="4"></td></tr>
<tr><td>Date:</td><td colspan="4"></td></tr>
</table>

<table>
<tr><td>Practical No.: 7</td><td colspan="2"></td><td colspan="2">Enrolment No.</td></tr>
<tr><td>Objective:</td><td colspan="4">To understand the concept of context aware programming.</td></tr>
<tr><td>Practical Exercise:</td><td colspan="4">Consider the following prototype of code.

<pre>def change(b, c, d):
    def a(x):
        return b(c(d(x)))
    return a</pre>
Perform the transformation of parameters b, c, d, in respective measurement unit such as meter to foot to inch. Consider base measurement as kilometre.

Ex: b(c(d(x)))
X=3

Output: 118110.24
Show context with each function call such as conversion to meter.</td></tr>
<tr><td>Pre-requisite:</td><td colspan="4">Currying and Monads.</td></tr>
<tr><td>Duration for Completion:</td><td>04</td><td colspan="3"></td></tr>
<tr><td>Reference to solve the problem:</td><td colspan="4">Functional Python Programming by Steven F. Lott , https://www.python-course.eu/currying_in_python.php</td></tr>
<tr><td>Solution must contain:</td><td colspan="4">Written description for the given problem and printout of output screen.</td></tr>
<tr><td colspan="5">Evaluation Parameters: 1. Implemented Monads, 2. Correct composition of functions, 3. Technical viva</td></tr>
<tr><td>Marks:</td><td>4</td><td>4</td><td>2</td></tr>
<tr><td>Secured by Student:</td><td></td><td></td><td></td></tr>
<tr><td>Teacher's Signature:</td><td colspan="3"></td></tr>
<tr><td>Date:</td><td colspan="3"></td></tr>
</table>

Practical No.: 8		**Enrolment No.**		
Objective:	To apply the concept of vectorization.			
Practical Exercise:	Solve the given practical problems using numpy library. 1. Consider n=10, k=5, m=7 and build and display two matrices of random numbers whose elements are less than 100. Order of matrixA(n, k) Order of matrixB(k, m) Display both matrix and describe that the matrix multiplication of matrixA and matrixB is possible or not? Now, take two matrices namely matrix_M and matrix_V of order (100, 80) and (80, 90) respectively. Elements of both the metrices are random number less than 1000. Perform the matrix multiplication of matrix M and V using normal Python for loop and also using vectorization concept. Make use of time module to compare the time taken by normal Python for loop and vectorization using numpy library. Display the time taken by both techniques for matrix multiplication. 2. Write a program to create 9×9 matrix and fill it with checkboard pattern. Sample checkboard pattern for 3×3 matrix. [0 1 0 1 0 1 0 1 0] 3. Write a function to extract all the contiguous 3×3 blocks from the 9×9 arbitrary matrix.			
Pre-requisite:	Basics of Numpy library			
Duration for Completion:	**04**			
Reference to solve the problem:	Functional Python Programming by Steven F. Lott , https://numpy.org/			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Correct solution, 2. Usage of Numpy functions, 3. Extraction using vectorization, 4. Technical viva*				
Marks:	**02**	**03**	**03**	**02**
Secured by Student:				
Teacher's Signature:				
Date:				

Practical No.: 9		**Enrolment No.**
Objective:	To analyse the data using Pandas data frame.	
Practical Exercise:	A. Consider the daily basis tweets samples collected from Twitter which contains high-frequency hashtag '#Omicron' for that download 'omicron.csv' from https://www.kaggle.com/gpreda/omicron-rising. Perform the asked queries and display output in proper format.	

1. Read the omicron.csv file using Pandas library and store the dataset using list and array data structure.
2. Find the username and location who is having highest followers.
3. Find all the tweets from 'United Kingdom'.
4. Count total users per location.
5. Find each locations highest users friend.
6. Find the average retweets of each country.
7. Find median and mode for user's friend and followers.
8. Display retweets and user name whose tweet is retweeted more than 300 times [use zip()]
9. Display top five tweets details based on retweets column. [use sorting]
10. Display all tweets contains hashtag Omicron.
11. Display all users with their source as iPhone and Android.
12. Display all the tweets only from verified user.
13. Display date wise average tweets from Canada region.
14. Find all tweets description contains the word 'COVID'
15. Create two data frames using the following two dictionaries and get the desired output.

```
IndiaLocation = {'location': ['Mumbai', 'Gurgaon', 'Bangalore', 'New Delhi'], 'tweets': [23845, 171995, 135925 , 71400]}
USLocation = {'location': ['Wisconsin', 'West Virginia', 'Ohaio', 'California'], 'tweets': [29995, 23600, 61500 , 58900]}
```

Desired output:

		location	tweets
India	0	Mumbai	23845
	1	Gurgaon	171995
	2	Bangalore	135925
	3	New Delhi	71400
United States	0	Wisconsin	29995
	1	West Virginia	23600
	2	Ohaio	61500
	3	California	58900

16. Create two data frames using the following two Dictionaries, Merge two data frames, and append the second data frame as a new column to the first data frame.

```
Loc_retweets = {'location': ['Mumbai', 'Gurgaon', 'Bangalore', 'New Delhi'], 'retweets': [23845, 17995, 135925 , 71400]}
```

```
Loc_favorites = {'location': ['Mumbai', 'Gurgaon', 'Bangalore',
'New Delhi'], 'favorites': [141, 80, 182 , 160]}
```

Desired output:

	location	retweets	favorites
0	Mumbai	23845	141
1	Gurgaon	17995	80
2	Bangalore	135925	182
3	New Delhi	71400	160

B. Perform the following in functional style.

1. Create a function which will take list of numbers as input and check that any of the number from inputed list is odd or not. [Use of existential quantifier is mandatory, Usage of map or list comprehension or lazy evaluation are appreciable.]

2. Create a function which will take a string as an input and check whether the inputted string does not contain any number. [Use of Universal quantifier is mandatory]

Pre-requisite:	Pandas data frame and iterables			
Duration for Completion:	**04**			
Reference to solve the problem:	https://pandas.pydata.org/pandas-docs/stable/getting_started/tutorials.html			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Usage of vectorised data structure 2. Usage of Iterable functions 3. Correct output 4. Technical viva*				
Marks:	**03**	**03**	**02**	**02**
Secured by Student:				
Teacher's Signature:				
Date:				

Practical No.: 10		**Enrolment No.**
Objective:	To utilize the Python built-in module as well as creation of custom module.	
Practical Exercise:	1. Consider "orderDetails.csv" file and display all bill in given format by creating custom template. Set a custom delimiter '#' for the placeholder variables by overriding the Template class. Desired output: **Dheeraj Sons** Bill No: 1 Order No: 10248 Customer Name: VINET **Product Name** **Price** **Quantity** Queso Cabrales 14 12 Singaporean Hokkien Fried Mee 9.8 10 Mozzarella di Giovanni 34.8 5 Total Amount: 58.6 Hint: make a list of dictionary contains item name, unit price and quantity. 2. Write a program to display following things in sub-directory of given path, proper validation required such as directory exist or not: i. Display details of file such as name, type and size ii. Display file details if the file size is 0. iii. Display summary of available files with count such as total image files, total pdf files and total text files. iv. Display details of file such as last access time, author and access mode. v. Display list of sub-directory available in given path. vi. Delete all the pdf files available in given path. 3. Create your own 'Validation' module and apply that module in Practical-4 to create a new bank account. For that modify the Practical-4 script and do as directed. Following points should be kept in mind while creating Validation module and *must display appropriate message for each invalid input*. i. While customer enter bank account number for deposit or withdrawal check whether the account number is of eight digits and it contains only digit. ii. Customer name only contains alphabets, no digit or special symbols are acceptable. iii. Date of birth should be in DD/MM/YYYY format only. iv. Valid values of gender are 'm', 'M', 'f', 'F', 't', or 'T' only. [Or student can think of alternate option for gender input] v. Address should be of maximum fifty characters long. vi. Contact number contains only digit and should be on length ten. vii. Email address contains '@' and '.' and support any email server, maximum length is fifty. viii. Valid PAN starting with five alphabets then four digits and number should end with one character. Ex: XXXXX1234X.	
Pre-requisite:	Basic knowledge of os, sys, string and re module.	

Duration for Completion:	**04**			
Reference to solve the problem:	https://docs.python.org/			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Correct script for asked functionality 2. Usage of in-built module 3. Creation and implementation of custom module 4. Technical Viva*				
Marks:	**2**	**3**	**3**	**2**
Secured by Student:				
Teacher's Signature:				
Date:				

<table>
<tr><td>Practical No.: 11</td><td colspan="2"></td><td>Enrolment No.</td></tr>
<tr><td>Objective:</td><td colspan="3">Lambda function and Object-oriented concepts in Python.</td></tr>
<tr><td>Practical Exercise:</td><td colspan="3">1. Write the following functions as lambda expression, and assign them to variables and call that variable to display the result.

i. Take one parameter and return its square.
ii. Take two parameters and return the square root of the sums of their squares.
iii. Take any number of parameters and return their average.
iv. Take a string parameter and return a string which contains the unique letters in the input string (in any order)

2. Write a python program using lambda expression to sort the order of programming in given list of dictionary based on its alphabet sequence. The expected result is as following:

<code>The assigned list of dictionaries :
[{'Name': 'Parimal', 'Programming': 'Julia', 'Year of Experience': 9}, {'Name': 'Mayur', 'Programming': 'C', 'Year of Experience': 4}, {'Name': 'Hiren', 'Programming': 'Python', 'Year of Experience': 6}]

Output:
[{'Name': 'Mayur', 'Programming': 'C', 'Year of Experience': 4}, {'Name': 'Parimal', 'Programming': 'Julia', 'Year of Experience': 9}, {'Name': 'Hiren', 'Programming': 'Python', 'Year of Experience': 6}]</code>

3. Rewrite the ‘Practical-4’ using object-oriented fundamentals according to given specification and test the program for at least five customers. [Use generator for creating account number.]

Create a Bank class with instance variable, class variable, accessor, mutator and other required methods to perform the given task, for that use the following information.

Bank class have,
List of variables: accountNo, Amount, IFSC, noOfCustomer
List of methods: openAccount(), getAccNo(), getAmount(), getIFSC(), setAmount(), withdraw(), deposit(), limit() to check for sufficient balance on the time of withdraw, customerCount(), and closeAccount()

Write the bifurcated list of this information into instance variable, class variable, instance method, class method and static method according to their nature.

[Student can introduce and use other methods then specified, but mandatory to provide the reason for using that method.]</td></tr>
<tr><td>Pre-requisite:</td><td colspan="3">Anonymous function, lazy evaluation and OOP fundamentals</td></tr>
<tr><td>Duration for Completion:</td><td>04</td><td colspan="2"></td></tr>
<tr><td>Reference to solve the problem:</td><td colspan="3">Steven F. Lott, Functional Python Programming Second Edition, Packt</td></tr>
</table>

Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Proper usage of lambda expression, 2. Correct identification of OOP features, 3. Correct implementation of Bank class with generator function, 4. Technical Viva*				
Marks:	**4**	**3**	**2**	**1**
Secured by Student:				
Teacher's Signature:				
Date:				

Practical No.: 12		**Enrolment No.**
Objective:	To implement the concept of recursion and reduction.	
Practical Exercise:	Consider the given Python script for Binary Search using iterative approach and convert it using Recursive approach.	

Write prerequisite script for taking input for the Binary search algorithm.

- Take length of list from user
- Take elements and store it in one list
- Sort the inputted list
- Take target element to be found

IterativeBinarySearch.py

```
#Define variables
start = 0
end = length-1
position = -1

while(start<=end):
 mid = (start+end) // 2
 if int_list[mid] == target:
   position = mid
   break
 #If not, check if lesser than mid element
 #Change range to start to mid-1, since less than mid
 elif target < int_list[mid]:
   end = mid-1
 #Check if lesser than mid element
 #Change range to mid+1 to end, since greater than mid
 elif target > int_list[mid]:
   start = mid+1

if position == -1:
   print('Element not in list')
else:
   print("Element found at position: "+ str(position+1))
```

Sample Output:

```
Enter length of list: 7

Enter element: 61

Enter element: 45

Enter element: 46

Enter element: 91

Enter element: 2

Enter element: 8

Enter element: 5
[2, 5, 8, 45, 46, 61, 91]
```

	`Enter target element: 8` `Element found at position: 3`			
Pre-requisite:	Recursion			
Duration for Completion:	**02**			
Reference to solve the problem:	Steven F. Lott, Functional Python Programming Second Edition, Packt			
Solution must contain:	Written description for the given problem and printout of output screen.			
Evaluation Parameters: *1. Correct implementation of Base condition, 2. Correct implementation of Recursive condition, 3. Take list of elements from user and sort the list before passing it to function, 4. Other cares taken related to Binary Search.*				
Marks:	**2**	**3**	**3**	**2**
Secured by Student:				
Teacher's Signature:				
Date:				