SANKALCHAND PATEL UNIVERSITY ।। अथातो ज्ञानजिज्ञासा ।।	**FACULTY OF COMPUTER SCIENCE** **Master of Computer Application (Integrated) (Sem-V)** In Effect from Academic Year 2023-24

Branch Name:	IMCA
Program Code:	CS301
Course Title	Advanced Python Practical
Course Code	1CS3010503P
Pre-requisite Course:	Basics of computer science including algorithms, data structure, probability theory

Course Objective:

The objectives of the course are to:

- To be able to understand the various regular expressions available in the Python programming language and apply them.
- To understand the advanced concepts of database programming, multithreading etc
- To be able to use different libraries like Pandas, NumPy,Natplotlib,SciPy etc.
- To be able to understand the concepts of data analytics.

Teaching Scheme (Hours per week)				**Evaluation Scheme (Marks)**				
				Theory		**Practical**		
Lecture	**Tutorial**	**Practical**	**Credit**	**University Assessment**	**Continuous Assessment**	**University Assessment**	**Continuous Assessment**	**Total**
-	-	3	3	-	-	25	25	50

Sample Practical List

1. Create Regular Expressions that
   a. Recognize following strings bit, but, bat, hit, hat or hut
   b. Match any pair of words separated by a single space, that is, first and last names.
   c. Match any word and single letter separated by a comma and single space, as in last name, first initial.
   d. Match simple Web domain names that begin with www. and end with a ".com" suffix; for example, www.yahoo.com. Extra Credit: If your regex also supports other high-level domain names, such as .edu, .net, etc. (for example,www.foothill.edu).
   e. Match a street address according to your local format (keep your regex general enough to match any number of street words, including the type designation). For example, American street addresses use the format: 1180 Bordeaux Drive. Make your regex flexible enough to support multi-word street names such as: 3120 De la Cruz Boulevard.
2. Create Regular Expressions That:
   a. Extract the complete timestamps from each line.
   b. Extract the complete e-mail address from each line.
   c. Extract only the months from the timestamps.
   d. Extract only the years from the timestamps.
   e. Extract only the time (HH:MM:SS) from the timestamps.
3. Create utility script to process telephone numbers such that
   a. Area codes (the first set of three-digits and the accompanying hyphen) are optional, that is, your regex should match both 800-555-1212 as well as just 555-1212.
   b. Either parenthesized or hyphenated area codes are supported, not to mention optional; make your regex match 800-555-1212, 555-1212, and also (800) 555-1212.
4. Create utility scripts when processing online data:
   a. HTML Generation. Given a list of links (and optional short description), whether user- provided on command-line, via input from another script, or from a database, generate a Web page (.html) that includes all links as hypertext anchors, which upon viewing in a Web browser, allows users to click those links and visit the corresponding site. If the short description is provided, use that as the hypertext instead of the URL.
   b. Tweet Scrub. Sometimes all you want to see is the plain text of a tweet as posted to the Twitter service by users. Create a function that takes a tweet and an optional "meta" flag defaulted False, and then returns a string of the scrubbed tweet, removing all the extraneous information, such as an "RT" notation for "retweet", a leading ., and all "#hashtags". If the meta flag is True, then also return a dict

containing the metadata. This can include a key "RT," whose value is a tuple of strings of users who retweeted the message, and/or a key "hashtags" with a tuple of the hashtags. If the values don't exist (empty tuples), then don't even bother creating a key-value entry for them.

c. Amazon Screen Scraper. Create a script that helps you to keep track of your favorite books and how they're doing on Amazon (or any other online bookseller that tracks book. rankings). For example, the Amazon link for any book is of the format, http://amazon.com/dp/ISBN (for example, http://amazon.com/dp/0132678209). You can then change the domain name to check out the equivalent rankings on Amazon sites in other countries, such as Germany (.de), France (.fr), Japan (.jp), China (.cn), and the UK (.co.uk). Use regular expressions or a markup parser, such as BeautifulSoup, lxml, or html5lib to parse the ranking, and then let the user pass in a commandline argument that specifies whether the output should be in plain text, perhaps for inclusion in an e-mail body, or formatted in HTML for Web consumption.

5. Process Result.txt file having data in the format (Format: EnrollmaentNumber, Name, Sub1_marks, Sub2_Marks, Sub3_marks,sub4_Marks) and generate analysis in following format
   a. Print Marksheet (Design your own format)
   b. Generate Student Summary (Enrollment Number, Name, Marks Obtained, Pass/Fail)
   c. Grade wise Summary
6. Process large mailbox.txt file having all email messages. Using regular expressions recognize all email addresses and URLs and save them into links.html. Use threads to segregate the conversion process.
7. Create a set of threads to count how many lines are there in a set of text files.
8. Create an extension or module library in Python to implement Calculator.
9. Create Database Application "Address Book" with options to add, modify, view and delete entry option.
10. Write a Python CGI script to display "Hello, World!" on a web page.
11. Create an HTML form to take a user's name and age. Write a Python CGI script to process the form and display the data.
12. Build an HTML form with two input fields for numbers and dropdown options for operations (+, -, *, /). Write a Python CGI script to perform the selected operation and display the result.
13. Create an HTML login form. Write a Python CGI script to check the username and password against predefined values.
14. Implement a basic session management system to track user data (e.g., login status) across multiple CGI scripts.
15. Create an HTML form to allow file uploads. Write a CGI script to save the uploaded file on the server and provide a download link.
16. Develop a CGI-based web app that connects to a database (e.g., MySQL or SQLite) to insert, update, and fetch records.

List of References:

1. Wesley J Chun, Core Python Applications Programming, 3rd Edition.Pearson
2. Michael Bowles, Machine Learning in Python, Essential techniques for predictive analysis, Wiley
3. Mark Pilgrim, Dive into Python: Python Novice to pro(source: http://diveintopython.org/.)
4. Alex Martelli, Python Cookbook, O'REILLY
5. Luke Sneeringer, Professional Python, WROX
6. Laura Cassell, Python Projects, WROX

Web Resources

1. http://docs.python.org/library/csv
2. http://docs.python.org/library/json
3. http://docs.python.org/library/ext
4. http://en.wikibooks.org/wiki/Python_Programming
5. http://learnpythonthehardway.org/
6. http://jason.org
7. Nosql-database.org
8. www.mongodb.org/
9. W3schools.com

Course Learning Outcomes (CLO): On completion of this course, the students will be able to:

CLO	Description	Bloom's Taxonomy Level
CLO1	Store and clean data with Pandas Data frames and can use NumPy, Matplotlib and Scipy.	2 Understanding 3 Applying 4 Analyzing
CLO2	Understand the concepts of data analytics	1 Remembering 2 Understanding
CL03	Create the regular expression in python code.	3 Applying 6 Creating
CLO4	Implement the multithreading concepts in python code.	3 Applying 4Analyzing 5 Evaluate 6 Creating
CLO5	Implement the concepts of database programming	3 Applying 4Analyzing 5 Evaluate 6 Creating
CLO6	Create the application using advanced python methodology.	5 Evaluate 6 Creating

Mapping of CLOs with POs & PSOs

Course Learning Outcomes	Program Outcomes (POs)												Program Specific Outcomes (PSOs)	
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CLO1	H	H	L		M		L		L	L	H	M	H	H
CLO2	M	H		L	L		L		M	L	L	M	M	
CLO3	M	H	H	L	L		L		L	L	M	M		M
CLO4	M	H	M	L	L		L		L	L	M	M	H	
CLO5	M	H	H	L	L		L		L	L	M	M		H
CLO6	M	H	H	L	L		L		L	L	M	M	H	H

H: High, M: Medium, L: Low