Course Title: Problem-Solving and Python Programming			
Course Code: UDS 102	**No. of credits:** 3	**L-T-P:** 20-10-30	**Learning hours:** 45
L: Lectures; **T:** Tutorials; **P:** Practicals			
Pre-requisite Course Code and Title (if any): None			
Department: Natural and Applied Sciences			
Course Coordinator: Dr. Adwitiya Sinha	**Course Instructor:**		
Contact Details:			
Course Type: Major	**Course Offered in:** Semester-2		

Course Description

The course begins with exploring diverse ways of computational approaches and applying them for problem-solving. The fundamentals include building blocks of algorithmic problem solving, pseudo code, flow charts, program flow diagrams, and infographics. The students will get acquainted of strategies for designing algorithms using iteration, recursion, function, and file handling. In the later phase, Python constructs would be introduced as a programming tool to develop problem-solving approaches. This will involve the basics of Python, control statements, data structures, lambda functions, modules, and packages. The course will also include visualizations, data, and code storytelling, and iPython for interactive computing. Overall, the course covers complete understanding of Python programming with primary focus on problem solving approaches to real-world cases.

Course Objectives

- To understand the fundamentals of algorithmic approaches to problem solving
- To learn Python programming essentials using conditionals, iterations, recursions
- To use functions and file handling in Python
- To utilize data structures in Python for data representation and manipulation
- To perform data visualization, file handling and interactive computing in Python

Course Content

Module	Topic	L	T	P
1	**Computational Problem Solving**			
	The objective of this module is to develop methodical approach to defining, framing, examining, and resolving problems through computational and algorithmic methods. This enables the decomposition of complex problems into sub-problems and providing rational solutions using computational thinking. The module will include the following topics: Computing fundamentals for problem solving, Identification of computational problems, Pseudo codes and algorithms, building blocks of a program (control statements, iteration, functions, recursion), designing flow chart, program flow diagrams, infographics with Canva, visual paradigms, Visme, etc.; instances of algorithmic problem solving; application to geoinformatics in data acquisition, spatial data science, and remote sensing tasks.	5	2	4
2	**Fundamentals of Python Programming**			

	This module provides insights on the fundamentals of Python programming and encompasses the essential principles and foundational concepts that serve as the building blocks for writing code in the Python programming language. After completing this module, the students will be able to perform effective programming in Python. Following will be covered in this context: Python basics, interpreter, code debugging, data types-integer, floating point, Boolean, string; variables, expressions, simple and conditional statements, input, and output statements, chained conditional statements, nested conditional statements, operators, precedence of operators, iterative constructs, while, for, break, continue, pass; use of single-line and multi-line comments	5	4	8
3	**Data Structures & Functional Programming in Python**			
	The purpose of this module is to combine the data structures with functional programming in Python to develop a powerful and expressive approach to solving complex problems. This will assist the learners to address wide range of computational problems with clarity, modularity, and scalability in Python code. The topics to be covered in this module include: Defining functions, actual and formal arguments, return statement, local and global scope of variables; List: list operations, slicing, list methods, Tuples: tuple assignment, tuple as return value; Dictionaries: operations and methods; sets, advanced list processing- list comprehension, sorting, searching, lambda expressions, lambda functions, regular expressions, arrays using Numpy, file handling (read, open, close), context managers, handling different file types (text, csv, json, etc.) , file opening modes, file operations, managing files with Pandas, scientific computing using SciPy	5	2	6
4	**Data Visualization & Interactive Python**			
	This module will connect the programming and problem-solving fundamentals with interactive and impactful data visualizations using Python. It will allow the students to present meaningful insights using graphical representations. The knowledge of dynamic visualizations will enhance the presentability of complex information in a comprehensible manner, especially for advanced topics and real-world applications. This module will cover following contents: Significance of data visualization, Python libraries for visualization, Matplotlib for Line, scatter, bar plots; customizing plots, multiple plots and subplots, time series plots, statistical visualization using Seaborn, heatmaps, correlation plots, interactive visualizations using Plotly, Pandas plotting, extraction of geospatial information using Selenium, Scrapy, BeautifulSoup, etc., geospatial data visualization with GeoPandas, Folium, Basemap; case study in geoinformatics using PySAL and Rasterio, 3-dimensional plots, interactive dashboards using Dash, data/code storytelling, iPython	5	2	12
	Total	**20**	**10**	**30**
Practical	Flow diagram tools like draw.io, Visio, etc., infographics with Canva, Visual Paradigm, Visme, etc.			4

	Python Programming, basic I/O constructs, simple and conditional statements, iteration, recursion			8
	File handling, list, tuple, set, dictionary, Practical with libraries, like Numpy, Pandas, SciPy, etc.			6
	Hands-on with data visualizations, higher dimensional graphical representations, Practical with libraries, like Plotly, Selenium, Scrapy, BeautifulSoup, GeoPandas, Folium, Basemap, PySAL, Rasterio, Dash, iPython; Minor project and case studies.			12
	Total			**30**

Evaluation criteria

- Minor Test 1: Written test [at the end of teaching of modules 1 and 2] -- 20%
- Minor Test 2: Written test [at the end of teaching of module 3] -- 20%
- Practical: Practical test [including modules 1, 2, and 3] -- 20%
- Minor Project: Project-based learning [at the end of teaching of module 4] -- 10%
- Major Test: Written test [at the end of the semester, full syllabus] -- 30%

Learning outcomes

By the end of the course, students will:

- Acquire a critical understanding of problem-solving techniques [module 1 and 2; minor test 1]
- Develop knowledge of building algorithms and programming with python [module 2 and 3; minor test 2; practical]
- Perform file handling, functional programming, data analysis [module 3 and 4; practical; major test]
- Perform data processing, programming, visualization, interactive computing [Module 1, 2, 3, and 4; Minor Project; Major Test]

Pedagogical approach

- The course critically evaluates the concepts of programming with Python through classroom discussions, lectures, tutorials, and project-based learning with real-world case studies.
- The course will allow learners to engage with enough hands-on sessions that will help in bridging the gap between theoretical understanding and real-world problem solving.
- The course will offer opportunities to explore how computational concepts and problem-solving techniques can address challenges in industry-relevant scenarios.

Reading Resources (* = compulsory readings)

- * Dromey, R.J. (2008). *How to Solve it by Computer*. University of Wollongong, Pearson Education, ISBN: 9788131705629, 442 pages.
- * Brown, M.C. (2001). *Python: The Complete Reference*. Osborne/McGraw-Hill, ISBN: 9780072127188, 691 pages.
- * Chen, D.Y. (2017). *Pandas for Everyone: Python Data Analysis.* Pearson Education, ISBN: 9780134547053, 416 pages.
- Lubanovic, B. (2014). *Introducing Python Modern Computing in Simple Packages, First Edition.* O'Reilly Media, ISBN: 9781449359362, 454 pages
- Chun, W.J. (2006). *Core Python Programming, Second Edition.* Pearson Education, ISBN: 9788131711880, 1137 pages
- Downey, A.B. (2015). *Think Python: How to Think like a Computer Scientist, Second Edition.* O’Reilly Publishers, ISBN: 9781491939413, 292 pages
- Kanetkar, Y., Kanetkar, A. (2020). *Let Us Python, Second Edition.* BPB Publications, ISBN: 9789389845037, 359 pages

Student Responsibilities

The students are required to come prepared with readings that would be given in the class. The students are required to participate in the class discussions.

Course Designed by:

- Dr. Adwitiya Sinha, Associate Professor, Department of Computer Science & Engineering and Information Technology, Jaypee Institute of Information Technology, Noida-62, Uttar Pradesh – 201309

Course Reviewers:

The course is reviewed by the following reviewers:

- Dr Satish Chand, Professor, School of Computer & Systems Sciences, Jawaharlal Nehru University, New Delhi.
- Dr Bijendra Kumar, Professor & Head (CSE), Department of Computer Science & Engineering, Netaji Subhas University of Technology, New Delhi