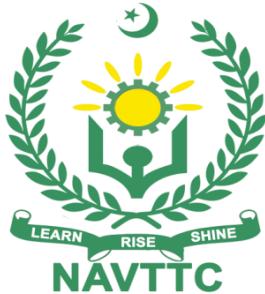Government of Pakistan

National Vocational and Technical Training Commission

Prime Minister Hunarmand Pakistan Program,
"Skills for All"

Course Contents/ Lesson Plan
Course Title: Advance Python Programming
Duration: 3 Months

Trainer Name	
Course Title	**Advance Python Programming**
Objective of Course	"Advance Python Programming" course is to equip individuals with employable skills and hands-on practice for Python programming and its applications. In line with the latest TIOBE Programming Community Index, which ranks Python among the top 10 popular programming languages, this course aims to provide comprehensive training in Python's general-purpose and high-level programming capabilities. Participants will learn to develop desktop GUI applications, websites, and web applications using Python. With its emphasis on simplicity and readability, Python enables developers to focus on core functionality while streamlining common programming tasks. By mastering advanced technologies and terminologies in Python, participants will gain the expertise needed to create efficient and maintainable tools and software in various domains. Through practical exercises, real-world projects, and a structured curriculum spanning 12 weeks, participants will enhance their Python programming skills and become proficient in leveraging Python's capabilities for software development and problem-solving. This course aims to empower individuals to excel in the dynamic field of programming and stay competitive in today's job market.
Course Execution Plan	Total Duration of Course: **3 Months (12 Weeks)**
	Class Hours: **4 Hours per day**
	Theory: 20% Practical: 80%
	Weekly Hours: **20 Hours Per week**
	Total Contact Hours: **240 Hours**

Learning Outcome of the Course	By the end of the course, students will have acquired a comprehensive skill set in advanced Python programming, empowering them to tackle challenging real-world problems and contribute effectively to software development projects. • Master program design, development, debugging, execution, and refactoring in Python. • Apply algorithmic thinking to solve problems effectively in Python. • Utilize Python Standard Library to resolve implementation challenges. • Understand the programmer's role in software development. • Create a standout programming portfolio for the job market
Companies Offering Jobs in the respective trade	• Google • Facebook • Amazon • Microsoft • Apple • IBM • Intel • Cisco • Oracle • NVIDIA • Freelancing These companies, among others, regularly seek professionals proficient in advanced Python programming skills for various positions ranging from software development to data analysis and machine learning.
Job Opportunities	In recent years, Python has emerged as a leading programming language globally, witnessing a surge in demand across industries. From mobile application development to web development, and from machine learning to data science, the industry seeks skilled Python developers. The ubiquitous presence of machine learning underscores the widespread adoption of Python. This trend not only presents lucrative earning prospects but also opens pathways for establishing a rewarding career in this dynamic field. Through this course, we aim to equip our youth with cutting-edge technical training in information technology. Additionally, the burgeoning market demand fosters opportunities for startup entrepreneurship, particularly in the designated roles mentioned above. • Python Developer • Software Engineer • Data Analyst • Machine Learning Engineer • Web Developer • Automation Engineer • GUI Developer • Data Scientist • Full Stack Developer • AI Engineer
No of Students	20-24

Learning Place	Labs
Instructiona lResources	These links will guide students to dedicated resources providing tutorials, courses, documentation, and practice platforms, all aimed at improving their proficiency in advanced Python programming. **Development Platform:** ● https://colab.research.google.com ● https://www.codeskulptor.org ● https://www.repl.it ● https://anaconda.com **Learning Material:** ● https://github.com ● https://docs.python.org ● https://w3schools.com ● https://tutorialspoint.com ● https://simpleisbetterthancomplex.com ● https://www.geeksforgeeks.org ● https://www.edx.org/course/introduction-to-computer-science-and-programming-7 ● https://www.codecademy.com/learn/learn-python-3 ● https://realpython.com/ ● https://pythoninstitute.org/

Scheduled Week	Module Title	Day	Learning Units	Home Assignment
Week 1/12	Basic Programming Concepts	Day 1	Introduction to Python - History, creators, significance. - Exploring Python's popularity and its applications in web development. - Understanding different versions of Python and minor differences. - Basic concepts: Variables, data types, and basic operations.	• **Task 1:** _Details may be seen at Annexure-I_
		Day 2	Installing Python 3.7+ version. - Setting up Anaconda distribution for Python. - Installing packages using PIP (Module Installations). - Basic concepts: Control structures (if, else, elif), loops (for, while).	
		Day 3	Setting up Virtual Environments for Python projects. - Overview of Python IDEs and Text Editors (Jupyter Notebooks, Pycharm Community Edition, Spider). - Basic concepts: Functions, modules, and libraries.	
		Day 4	Introduction to Python Standard Library. - Exploring commonly used modules in Python Standard Library (e.g., os, sys, math). - Basic concepts: Exception handling, file handling.	
		Day 5	Review of the week's concepts and practical exercises. - Basic concepts: Object-oriented programming (classes, objects, inheritance). - Developing a simple Python project incorporating concepts learned during the week. - Weekly Task: Develop a small command-line application using Python that incorporates concepts covered during the week.	
Week 2/12	Introduction to Advanced Python Programming	Day 1	Basic Syntax (No Semicolon, indentation, No parenthesis). - Pep8. - Keywords. - Variables & Literals.	• **Task 2:** _Details may be seen at Annexure-I_
		Day 2	Starting out with Expressions. - Basic Types. - Working with variables.	
		Day 3	Arithmetic Operators: Unary, Binary, +, -, x, /, //, %, **. - Percentage example. - Ratio example. - More Examples of simple calculations.	
		Day 4	Operators precedence. - Variables type conversion/casting.	
		Day 5	Introduction to Data Structures: List, Dictionary, Tuple, Sets.	
Week 3/12	Introduction to Debugging & User Interactions & Strings	Day 1	Syntax Errors - Runtime Errors - Semantic Errors - How to locate and resolve errors Interactions:	• **Task 3:** _Details may be seen at Annexure-I_
		Day 2	Output/Print (Constants, variables) - Input	
		Day 3	Input - Output - Concat	
		Day 4	Practical exercises on identifying and resolving syntax, runtime, and semantic errors in Python code.	

		Day 5	Practical exercises on using output/print statements to display constants and variables, and accepting input from users. Introduction to string manipulation and concatenation.	
Week 4/12	Advance Strings	Day 1	Introduction to Functions: - Syntax - Calling methods - Arguments - Return - How they work.	• **Task 4:** _Details may be seen at Annexure-I_
		Day 2	Types Mutation Methods: - Upper - Isupper - Lower - Islower - Count - Strip - Replace - Join - Split - Substring - Index - Negative index	
		Day 3	Practical exercises on defining and calling functions in Python.	
		Day 4	Practical exercises on using formatting methods such as upper, lower, count, strip, replace, join, split, substring, index, and negative index.	
		Day 5	Consolidation of the week's concepts through practical exercises and review.	
Week 5/12	List/Dictionary Data type & Methods	Day 1	List Methods: - Append - Pop - Prepend - Sort - Count - Index (+ve and -ve) - Insert - Remove	• **Task 5:** _Details may be seen at Annexure-I_
		Day 2	List comprehension	
		Day 3	Practical exercises on using list methods such as append, pop, sort, count, index, insert, remove, and list comprehension.	
		Day 4	Dictionary Methods: - Keys - Values - Items - Get - Merging - Pop - Clear - Copy	
		Day 5	Practical exercises on using dictionary methods such as keys, values, items, get, merging, pop, clear, copy, and dictionary comprehension. Consolidation of the week's concepts through practical exercises and review.	
Week 6/12	Tuple, Sets, Flow Control & Function	Day 1	uple: - Definition - Builtin Methods Sets: - Definition - Builtin Methods - Frozen Sets - Mutation	• **Task 6:** _Details may be seen at Annexure-I_
		Day 2	Conditions: - Simple - Multiple - Nesting - Logical Operators (>, <, ==, <, is, >=, <=)	
		Day 3	Loops: - While - For - For in - Nesting - Range - Break - Continue - Over List - Over String	
		Day 4	Nesting of loops & conditions Exception handling: - Try - Except	
		Day 5	Syntax: - Define - Calling Benefits: - Reuse - Code separation Arguments: - Default Arguments - Optional Arguments - Pass by Value & Reference Recursive Functions Lambda Functions Variable Scope (Global, local)	

Week 7/12	Mini Project, Employable Project/Assignment (5 weeks i.e. 7-12) in addition of regular classes.	Day 1	● Guidelines to the Trainees for selection of students employable project like final year project (FYP) ● Assign Independent project to each Trainee ● A project based on trainee's aptitude and acquired skills. ● Designed by keeping in view the emerging trends in the local market as well as across the globe. ● The project idea may be based on Entrepreneur. ● Leading to the successful employment. ● The duration of the project will be 6 weeks ● Ideas may be generated via different sites such as: https://1000projects.org/ https://nevonprojects.com/ https://www.freestudentprojects.com / https://technofizi.net/best-computer- science-and-engineering-cse-project- topics-ideas-for-students/ • Final viva/assessment will be conducted on project assignments. • At the end of session the project will be presented in skills competition • The skill competition will be conducted on zonal, regional and National level. • The project will be presented in front of Industrialists for commercialization • The best business idea will be placed in NAVTTC business incubation center for commercialization. OR On job training for 2 weeks: • Aims to provide 2 weeks industrial training to the Trainees as part of overall training program • Ideal for the manufacturing trades • As an alternate to the projects that involve expensive equipment • Focuses on increasing Trainee's motivation, productivity, efficiency and quick learning approach.	● **Task 7:** _Details may be seen at Annexure-I_
			Introduction to Project Examples: - Overview of Paper Scissors Rock Game.	
		Day 2	All Temperature Calculator: - Design and implementation.	
		Day 3	Unit Conversion System: - Design and implementation	
		Day 4	Number Guessing Game: - Design and implementation.	
		Day 5	Marks Grade/Average Calculations: - Design and implementation.	

Week 8/12	Packages/Modules &Object-Oriented Programming	Day 1	Introduction to Packages: - Examples (math, csv, os, sys).	• **Task 8:** _Details may be seen at Annexure-I_
		Day 2	Modules: - init .py file - Import: Start of file and in functions/blocks - Circular import.	
		Day 3	Classes and Objects: - Class - Object - Static Accessors - Private ('_') - Public Self Properties.	
		Day 4	Methods: - Constructor (init ()).	
		Day 5	Practical exercises on working with packages, modules, classes, and objects in Python.	
Week 9/12	Advance OOP	Day 1	Introduction to Inheritance: - Why inheritance? - Benefits of inheritance. - Single and multiple inheritance.	• **Task 9:** _Details may be seen at Annexure-I_
		Day 2	Overriding: - Method overriding.	
		Day 3	Polymorphism: - Method overriding for polymorphism.	
		Day 4	Composition: - Understanding composition in object-oriented programming.	
		Day 5	Operator Overloading: - Introduction to operator overloading.	
Week 10/12	I/O Operations	Day 1	File Handling: - Open - Modes (w, w+, wb, r) - With Keyword	• **Task 10:** _Details may be seen at Annexure-I_
		Day 2	File Types: - TXT - CSV: - Reader - Dict reader - Writer - Writerow - Writerows - JSON - XML	
		Day 3	Practical exercises on file handling operations such as opening, reading, writing, and closing files in various formats (TXT, CSV, JSON, XML)	
		Day 4	Array: - Array of array - Array of dicts - Array of tuples - Dictionary: - Array as values - Tuple as values - Dictionary as values	
		Day 5	Practical exercises on working with nested data structures such as arrays and dictionaries in Python.	
Week 11/12	Python Applications and Frameworks	Day 1	Desktop Applications: - Introduction to QT - Introduction to PyGUI - Introduction to Tkinter	• **Task 11:** _Details may be seen at Annexure-I_
		Day 2	Web Applications: - Introduction to Flask - Introduction to Django	
		Day 3	Introduction to Machine Learning, Artificial Intelligence, and Data Science: - Overview of PyTorch - Overview of Pandas - Overview of NumPy - Overview of SciPy - Overview of TensorFlow	
		Day 4	Practical exercises on developing desktop applications using QT, PyGUI, and Tkinter.	

		Day 5	Practical exercises on developing web applications using Flask and Django.	
Week 12/12	Final Project Progress & Evaluation		• Job Market Searching • Self-employment • Freelancing sites • Introduction • Fundamentals of Business Development • Entrepreneurship • Startup Funding • Business Incubation and Acceleration • Business Value Statement • Business Model Canvas • Sales and Marketing Strategies • How to Reach Customers and Engage CxOs • Stakeholders Power Grid • RACI Model, SWOT Analysis, PEST Analysis • SMART Objectives • OKRs • Cost Management (OPEX, CAPEX, ROCE etc.) • Final Assessment	• **Task 12:** _Details may be seen at Annexure-I_

List of Machinery / Equipment

  1. **Software List**

Sr. No	Software Name
1.	MS Office(Installed on each PC)
2.	Operating System (Windows, Linux or other Operating Systems)
3.	Programming Languages including PyCharm, Notebook
4	Database Management System (Optional): • SQLite • PostgreSQL • MySQL • MongoDB

  2. **Minimum Qualification of Teachers / Instructor**

The qualification of teachers / instructor of this course should be a minimum of bachelors inComputer science, / Software Engineering/ Statistics / Networks / Artificial Intelligence / Data Science with minimum 3 years of development experience in relevant trade.

  3. **Eligibility for the trainees:**

Bachelor's of Computers Science / Software Engineering/ Statistics / Networks / Artificial Intelligence / Data Science etc

  4. **Supportive Notes**

Teaching Learning Material

Books Name	Author
Python Crash Course	Eric Matthews
Learn Python the Hard Way	Zed A. Shaw
Python Programming: An Introduction to Computer Science	John Zelle

  5. **List of Machinery/Equipments**

Sr. No	Name of item as per curriculum	Quantity physically available at the training location
1	Computers Minimum Core-i7 8th Generation ● LCD Display 17" with built in speakers	25
2	DSL Internet Connection (Minimum 5 MB)	Available on every PC

3	**Accessories/Devices** ● Connectors ● Multimedia ● Printer (NW printer) ● Audio/visual aid ● White Board ● Pin Board ● Flip Chart Board ● Hard copy of Training Material ● Mobile Phones	25 each
4	**Wires, data cables, power plugs, powersupply**	For every PC
5	**UPS**	Available
6	**Generator / Solar Backup**	Available
7	**Air Conditioner (2 Tons)**	Available

Annexure # 01

Tasks for Certificate Advance Python Programming

Task No.	Description	Week
1.	Develop a small command-line application using Python that incorporates concepts covered during the week.	Week 1
2.	Write Python scripts to practice the following concepts: 1. Create variables and literals. 2. Implement basic arithmetic operations. 3. Perform conversions between variable types. etc.	Week 2
3.	Write Python scripts that include intentional syntax, runtime, and semantic errors, and practice debugging them.	Week 3
4.	Write Python scripts to demonstrate the use of functions, including defining functions, passing arguments, and returning values.	Week 4
5.	Implement Python scripts that utilize list and dictionary methods.	Week 5
6.	Write Python scripts to practice file handling operations for different file types (TXT, CSV, JSON, XML).	Week 6
7.	Choose one of the project examples discussed during the week and implement it in Python.	Week 7
8.	Implement a Python project that demonstrates inheritance, polymorphism, composition, and operator overloading concepts.	Week 8
9.	Implement a Python project that manipulates nested data structures such as arrays and dictionaries.	Week 9
10.	Write Python scripts that demonstrate file handling operations for different file types (TXT, CSV, JSON, XML) and manipulate nested data structures.	Week 10
11.	Implement a small project using desktop or web applications frameworks introduced during the week.	Week 11
12.	Develop a business proposal or plan for a startup idea or project.	Week 12