1. Course Code

2203

2. Course Title

F30e: Fundamentals of Computer Programming Python

3. Teacher

HAMIDULLAH, Sokout

4. Term

Fall 2

5. Course Requirements (Courses / Knowledge for this course) and Important Information

Fundamentals of Computer Systems (both courses can be taken concurrently)

6. Course Overview and Objectives

Programming is the foundation of every other subject in ICT. By becoming proficient in programming, students will be able to actively participate in projects involving system creation. Programming is also necessary for testing ideas, constructing and maintaining networks and servers, and in many other areas.
The course first reviews the fundamentals of procedural programming through experimental exploration, using the dynamic, interactive, object-oriented Python language. This course is the pave for Data Science and AI.

7. Course Outline

1. Course orientation, Python Objects and Data Structure Basics
2. Python Data Structure Advanced
3. Python Statements
4. Exercises
5. File Handling
6. Methods and Functions
7. Projects (Problem Sovling)
8. Exercises
9. Object Oriented Programming
10. Exercises
11. Errors and Exceptions Handling + Web Scraping
12. Python Database (MySQL) + Python Modules
13. Python GUI (Django + Gradio)
14. Exercises
15. Student Final Project
16. Presentation/ Reflection

8. Textbooks (Required Books for this course)

For this course, we prefer to use Jupyter Notebook. Jupyter Notebook Server available in case students have any difficulties with their own PCs

9. Reference Books (optional books for further study)

Python Crash Course, 2nd Edition: A Hands-On, Project-Based Introduction to Programming
Author: Eric Matthes
Publisher: No Starch Press
Online Resources

10. Course Goals (Attainment Targets)

(1) Become able to read, understand, and modify programs written in Python.
(2) Apply Control Structures and Functions
(3) Apply Object-Oriented Programming Concepts
(4) Perform File Handling and Input/Output Operations
(5) Develop Simple Applications and Scripts
(6)
(7)
(8)

11. Correspondence relationship between Educational goals and Course goals

Educational goals of the school			Course Goals
High level ICT skills	Basic academic skills		(1) (2) (3) (4) (5)
	Specialized knowledge and literacy		(3) (4) (5)
Human skill (Tankyu skill)	Ability to continually improve own strengths		(5)
	Ability to discover and resolve the problem in society	Problem setting	(5)
		Hypothesis planning	
		Hypothesis testing	
		Practice	(1) (2) (3) (4) (5)
	Fundamental Competencies for Working Persons	Ability to step forward	(5)
		Ability to think through	(5)
		Ability to work in a team	(1) (2) (3) (4) (5)
Professional ethics			

12. Evaluation

Goals	Evaluation method & point allocation					
	Examination	Quiz	Reports	Presentation	Deliverables	Other
(1)		○		○	○	○
(2)		○		○	○	○
(3)		○		○	○	○
(4)		○		○	○	○
(5)				○	○	○
(6)						
(7)						
(8)						
Allocation		20		30	40	10

13. Evaluation Criteria

Examination	
Quiz	Related weeks tasks will be given to students to solve in order to evaluate the understanding of students and motivate them for further learning. Expected codes should be without errors and original.
Reports	
Presentation	In the final presentation, students will be asked develope a small application using Python. The evaluation will be based on participation in the group work, presentation, the relavance of the argument, time managment and relation to the course contents. Meanwhile, its recommended to have specific instructions for code implementation.
Deliverables	1. Individual or group assignment will be assigned for the students with focus on learning goals (1,2 and 3). The evaluation will be based on how the students understand the exercises and participation. 2. The results required by the exercise can be achieved.
Other	Class Exercises Participation and Team Collaboration

14. Active Learning

Hourly percentage of active learning within the whole class time		60%
1	Active learning such as problem solving assignment using the knowledge and skills acquired in class.	Sometimes
2	Active learning such as group works and discussions.	All the time
3	Outcome presentations and feedbacks.	All the time
4	Students actively make decisions on how the class should be conducted.	Not at all

15. Notes

16. Course plan

(Notice) This plan is tentative and might be changed at the time of delivery

Lesson 1: Python Objects and Data Structures (Basic & Advanced) **Lecture + Exercises**

What is programming? We consider solving everyday tasks, first by natural language, then by writing exact and detailed instructions. Students learn the basics of the Python language through a few simple exercises and use it to test their understanding of the elements of programming principles.

Lesson 2: Python Objects and Data Structures (Basic & Advanced) **Lecture + Exercises**

Learn how to work with Python's built-in data types such as strings, numbers, lists, tuples, sets, and dictionaries.

1. Strings
2. Numbers
1. Lists
2. Dictionaries
3. Tuples
4. Sets

Lesson 3: Python Statements **Lecture + Exercises**

Explore conditional statements (if, elif, else) and loops (for, while) to control the flow of programs. Understand indentation, logical operators, and writing clear, structured code.

Lesson 4: Python Statements Exercises **Exercises**

Summarize the previous knowledge and practice comprehensively.

Lesson 5: File Handling **Lecture + Exercises**

Understand how to read from and write to files using Python. This includes handling text and CSV files, file modes, and best practices for managing file I/O operations.

Lesson 6: Methods and Functions **Lecture + Exercises**

Learn to define and use reusable code blocks with functions. This includes parameters, return values, and built-in vs. user-defined functions.

1. Built in Methods in Python
2. User define Methods

Lesson 7: Project-1 **Exercises**

Summarize the previous knowledge and practice comprehensively with different problems

Lesson 8: Exercises **Exercises**

Summarize the previous knowledge and practice comprehensively with different problems

Lesson 9: Object Oriented Programming **Lecture + Exercises**

Introduce core OOP principles such as classes, objects, inheritance. Students will design modular, maintainable programs using object-oriented design.

Lesson 10: Object Oriented Programming Exercises **Exercises**

Summarize the previous knowledge and practice comprehensively.

Lesson 11: Errors and Exceptions Handling + Web Scraping **Lecture + Exercises**

Learn to handle runtime errors using try-except blocks, raising exceptions, and using finally. Students will also explore basic web scraping using libraries like requests and BeautifulSoup.

1. Errors and Exeptions Handling with Python
2. Web Scaping concept and implementation with Python

Lesson 12: Python Database (MySQL) **Lecture + Exercises**

Introduce database concepts and show how to connect to and interact with databases using MySQL. Topics include CRUD operations and executing SQL queries in Python.

Introduce the fundamental packages for scientific computing with Python: Numpy, and Pandas.

Lesson 13: Python GUI (Django + Gradio) **Lecture + Exercises**

Learn to create simple graphical user interfaces using libraries like Gradio. Students will build interactive applications with input fields, buttons, and event-driven programming.

The most common frameworks (Django) will be also discussed, as well as Gradio which is an open source python package that can be used to quickly build a demo or web application for AI based solutions

Lesson 14: Python GUI (Django + Gradio) Exercises **Exercises**

Summarize the previous knowledge and practice comprehensively.

Lesson 15-16: Final Project Presentation/ Reflection **Presentation**

Presentation and discussion