**《 Course Syllabus》**

Course code：IS192

Course Name：Python

Course type：Required course

Prerequisite courses: None

Applicable majors: Network Engineering (China-US Cooperative Education)

College: Information and Mechanical and Electrical Engineering College

Total class hours: 48 hours

Credits:4.0

**Ⅰ、Course teaching objectives and requirements for cultivating students' abilities**

1. **Curriculum, teaching objectives, and main tasks**

This course is an introductory course which exposes students to the world of programming using python as a programming language. In Part I the course covers all the fundamental concepts in python programming starting from a basic “Hello World!” code to variables, lists, if statements, dictionaries, while loops, user input, functions, classes (OOP), handle errors, write tests for your code. In Part II students will learn about projects using pygame and other packages, data visualization using Matplotlib and Plotly, access data from online sources and visualize it, data science, and web applications using Django. By learning python, and its data structures and algorithms, students can easily switch over to any other language in future.

1. **Basic requirements for cultivating students' abilities**

Student will be expected to complete homework for every class. Class exercises to be completed in groups and individually will be given regularly to be completed with the assistance of the teacher in the classroom as needed. Weekly notes will be shared with students via WeChat on the class group chat.

**Ⅱ、The teaching content and learning objectives of the course**

**1．Theoretical teaching**

**Chapter 1**

Class hours: 10

Main Content: Where to Start

Teaching requirements:**about the VS Code editor…4, python on Linux…8, running hello\_world.py…10, troubleshooting…10, running python programs from a terminal on macOS and Linux…12**

Key points: python intro, install python, install text editor, run snippets of python in a terminal session, run your first program.

Difficult points: **Troubleshooting…10, Exercise 1-3…13**

Teaching methods: Class Lecture

**Chapter 2**

Class hours: 10

Main Content: Variables and Simple Data Types

Teaching requirements: **variables are labels…18, avoiding syntax errors with strings…24, constants…28, what kinds of comments should you write…29, the zen of python…30**

Key points: use descriptive variable names and resolve name errors and syntax errors, how to display strings using lowercase, uppercase, and title case, use whitespace to organize output neatly, remove unneeded elements from a string, work with integers and floats, learn some ways to work with numerical data, write explanatory comments, learn the philosophy of keeping code simple.

Difficult points: **Exercise 2-2: Simple Messages…19, Exercise 2-8: File Extensions…25, Exercise 2-10: Favorite Number…29, Exercise 2-11: Adding Comments…30, Exercise 2-12: Zen of Python…31.**

Teaching methods: Class Lecture

**Chapter 3**

Class hours: 10

Main Content: Introducing Lists

Teaching requirements: **using individual values from a list…35, removing elements from a list…38, finding the length of a list…44, avoiding index errors when working with lists…46**

Key points: Lists, working with elements in a list, define a list and add and remove its elements, sort lists permanently (and temporarily for display), find the length of a list, how to avoid index errors.

Difficult points: **Exercise 3-3: Your Own List…36, Exercise 3-7: Shrinking Guest List…42, Exercise 3-11: Intentional Error…47**

Teaching methods: Class Lecture

**Chapter 4**

Class hours: 10

Main Content: Working with Lists

Teaching requirements:**doing something after a for loop…52, forgetting the colon…55, list comprehensions…59, copying a list…63, writing over a tuple…67, other style guidelines…69**

Key points: how to work efficiently with elements in a list, work through a list using for loop, indentation, avoiding indentation errors, numerical lists, operations on numerical lists, slice a list to work on a list subset, copy lists properly using a slice, tuples, how to style complex code for readability.

Difficult points: **Exercise 4-2: Animals…56, Exercise 4-9: Cube Comprehension…60, Exercise 4-12: More Loops…65, Exercise 4-13: Buffet…67,**

**Exercise 4-15: Code Review…70**

Teaching methods: Class Lecture

**Chapter 5**

Class hours: 10

Main Content: if Statements

Teaching requirements:**boolean expressions…77, testing multiple conditions…82, using multiple lists…87, styling your if statements…89**

Key points: write conditional tests, if statements, if-else chains, if-elif-else chains, identify particular conditions to test and know when those conditions have been met, handle certain items in a list differently than all other items while continuing to utlize the efficiency of a for loop, revisit python style recommendations to ensure increasingly complex code is still relatively easy to read and understand..

Difficult points: **Exercise 5-2: More Conditional Tests…78, Exercise 5-7: Favorite Fruit…85, Exercise 5-11: Ordinal Numbers…88, Exercise 5-13: Your Ideas…89**

Teaching methods: Class Lecture

**Chapter 6**

Class hours: 10 hours

Main Content: Dictionaries

Teaching requirements:**using get() to access values…97, looping through all values in a dictionary…103, a dictionary in a dictionary…110**

Key points: define a dictionary, work the information in a dictionary, access and modify individual elements in a dictionary, loop through all information in a dictionary, loop through a dictionary’s key-value pairs, its keys, and its values, nest multiple dictionaries in a list, nest lists in a dictionary, nest a dictionary inside a dictionary.

Difficult points: **Exercise 6-3: Glossary…99, Exercise 6-6: Polling…105, Exercise 6-12: Extensions…111**

Teaching methods: Class Lecture

**Chapter 7**

Class hours: 10 hours

Main Content: User Input and While Loops

Teaching requirements: **the modulo operator…116, avoiding infinite loops…122, filling a dictionary with user input…125**

Key points: use input(), work with text and numerical inputs, use while loops to run programs, active flag, break statement, continue statement, use while loop to move items from one list to another, remove all instances of a value from a list, use while loops with dictionaries.

Difficult points: **Exercise 7-3: Multiples of Ten…117, Exercise 7-7: Infinity…123,**

**Exercise 7-10: Dream Vacation…127**

Teaching methods: Class Lecture

**Chapter 8**

Class hours: 10 hours

Main Content: Functions

Teaching requirements:**arguments and parameters…131, avoiding argument errors…136, using a function with a while loop…140, preventing a function from modifying a list…145, using arbitrary keyword arguments…148, importing all functions in a module…152, styling functions…153**

Key points: write functions to pass arguments, use positional and keyword arguments, accept an arbitrary number of arguments, use functions to display output and return values, use functions with lists, dictionaries, if statements, and while loops, store functions in modules, style functions so they are asy to read.

Difficult points: **Exercise 8-2: Favorite Book…131, Exercise 8-5: Cities…137, Exercise 8-8: User albums…142, Exercise 8-11: Archived Messages…146, Exercise 8- 14: Cars…149, Exercise 8-17: Styling Functions…154**

Teaching methods:Class Lecture

**Chapter 9**

Class hours: 10 hours

Main Content: Classes

Teaching requirements:**making an instance from a class…159, modifying attribute values…164, modeling real world objects…172,**

Key points: write classes, store information in a class using attributes, write methods that give classes behavior, \_init\_() methods that create instances from the class with desired attributes, modify attributes of an instance through methods, use inheritance to simplify creation of classes, use instances of one class as attributes in another class to keep each class simple.

Difficult points: **Exercise 9-3: Users…162, Exercise 9-5: Login Attempts…167, Exercise 9-9: Battery Upgrade…173, Exercise 9-12: Multiple Models…179, Exercise 9- 16: Python module of the week…180**

Teaching methods: Class Lecture

**Chapter 10**

Class hours: 10 hours

Main Content: Files and Exceptions

Teaching requirements: **is your birthday contained in pi?…189, writing multiple lines…191, deciding which errors to report…199, refactoring…204**

Key points: how to work with files, read the entire contents of a file, work through the contents one line at a time, write to a file, handle exceptions, store python data structures to save user input avoiding their need to start over each time they run a program.

Difficult points: **Exercise 10-3: Simpler Code…190, Exercise 10-5: Guest Book…192, Exercise 10-10: Common Words…200, Exercise 10-14: Verify User…206**

Teaching methods: Class Lecture

**Chapter 11**

Class hours: 10 hours

Main Content: Testing your Code

Teaching requirements:**installing pytest…211, adding new tests…216, using fixtures…221**

Key points:write tests for functions and classes using tools in the pytest module, write test functions that verify specific behaviors your functions and classes should exhibit, use fixtures to efficiently create resources that can be used in multiple test functions in a test file, test critical behaviors of your functions and classes, make improvements to your code, respond to failed tests (to avoid responding to angry users later), play around with tests, but don’t aim for full coverage.

Difficult points: **Exercise 11-2: Population…217, Exercise 11-3: Employee…223**

Teaching methods: Class Lecture

**Chapter 12**

Class hours: 10 hours

Main Content: A Ship That Fires Bullets

Teaching requirements:**creating a settings class…232, drawing the ship to the screen…235, the \_update\_screen() method…237, running the game in fullscreen mode…245, ship.py…246, creating the \_update\_bullets() method…252**

Key points: make a plan for a game, basic structure of a game in pygame, set background color and store settings in a separate class to adjust them more easy, draw an image to the screen, give player control over the movement of game elements, create elements that move on their own, delete objects that are no longer needed, refactor code in a project on a regular basis to facilitate ongoing development.

Difficult points: **Exercise 12-2: Game Character…238, Exercise 12-5: Keys…246, Exercise 12-16: Sideways Shooter…253**

Teaching methods: Class Lecture

**Chapter 13**

Class hours: 10 hours

Main Content: Aliens!

Teaching requirements:creating an instance of the alien…257, adding rows…261, dropping the fleet and changing direction…265, refactoring \_update\_bullets()…269, identifying when parts of the game should run…275

Key points: add a large number of identical elements to a game, use nested loops to create a grid of elements, make a large set of game elements move by calling each element’s update() method, control direction of objects on a screen, detect and respond to collisions, track statistics in a game, use a game\_active flag to determine when the game is over.

Difficult points: **Exercise 13-2: Better Stars…263, Exercise 13-4: Steady Rain…266, Exercise 13-5: Sideways Shooter Part 2…270, Exercise 13-6: Game Over…274**

Teaching methods: Class Lecture

**Chapter 14**

Class hours: 10 hours

Main Content: Scoring

Teaching requirements:**hiding the mouse cursor…282, resetting the speed…285, displaying the number of ships…296**

Key points: implement a Play button to start a new game, detect mouse events, hide cursor in the active game, create other buttons like a Help button, modify the speed of a game as it progresses, implement a progressive scoring system, display information in textual and nontextual ways.

Difficult points: **Exercise 14-2: Target Practice…283, Exercise 14-4: Difficulty Levels, Exercise 14-8: Sideways Shooter, Final version**

Teaching methods: Class Lecture

**Chapter 15**

Class hours: 10 hours

Main Content: Generating Data

Teaching requirements:**saving your plots automatically…311, styling the walk…315, saving figures…327**

Key points: generate datasets and create visualizations of that data, create simple plots with Matplotlib, use a scatter plot to explore random walks, create a histogram with Plotly and use it to explore the results of rolling dice of different sizes, keep an eye out for situations you might be able to model with code, identify news media using similar methods to your projects.

Difficult points: **Exercise 15-2: Colored Cubes…311, Exercise 15-5: Refactoring…319, Exercise 15-10: Practicing with Both Libraries…328**

Teaching methods: Class Lecture

**Chapter 16**

Class hours: 10 hours

Main Content: Downloading Data

Teaching requirements:**downloading your own data…341, mapping global datasets: GeoJSON format…342**

Key points: how to work with real world datasets, process CSV and GeoJSON files, extract data to focus on, use historical weather data with Matplotlib including how to use the datetime module and how to plot multiple data series on one chart, plot geographical data on a world map in Plotly, customize the style of the map.

Difficult points: **Exercise 16-5: Explore…342, Exercise 16-9: World Fires…352**

Teaching methods: Class Lecture

**Chapter 17**

Class hours: 10 hours

Main Content: Working with APIs

Teaching requirements:**Monitoring API rate limits…362, more about Plotly and the Github API…368**

Key points: use APIs to write self-contained programs that automatically gather data and use it to create visualizations, use GitHub API to explore projects on Github, look briefly at the Hacker News API, use the Requests package to automatically issue an API call and process results of that call, use some Plotly settings to further customize the appearance of generated charts.

Difficult points: **Monitoring API Rate Limits…362, more about Plotly and the GitHub API…368, Exercise 17-4: Further exploration…372**

Teaching methods: Class Lecture

**Chapter 18**

Class hours: 10 hours

Main Content: Getting Started with Django

Teaching requirements: **viewing the project…377, the Django shell…386, writing a template…390, individual topic pages…397**

Key points: build a simple web app using the Django framework, install Django to a virtual environment, set up a project, check the project is set up correctly, set up an app, define models to represent the data for your app, migrate databases using Django after a change to a model, create a superuser for the admin site, use the admin site to enter some initial data.

Difficult points: **Exercise 18-1: New Projects…378, Exercise 18-4: Pizzeria…388, Exercise 18-6: Pizzeria Home Page…392, Exercise 18-8: Pizzeria Pages…400**

Teaching methods: Class Lecture

**Chapter 19**

Class hours: 10 hours

Main Content: User Accounts

Teaching requirements: **editing entries…412, the registration page…420, associating new topics with the current user…429**

Key points: use forms to add new topics and entries and edit existing entries, implement user accounts, give existing users ability to log in and out, use Django’s default UserCreationForm to let people create new accounts, build a simple user authentication and registration system and then restrict access to logged-in users for certain pages using @login\_required decorator, assign data to specific users through a foreign key relationship, migrate the database when the migration requires to specify some default data, modify the view function so user can see only data that belongs to them, retrieve appropriate data using the filter() method, compare the owner of the requested data to the currently logged-in user.

Difficult points: **Exercise 19-1: Blog…415, Exercise 19-2: Blog Accounts…423, Exercise 19-5 Protected Blog**

Teaching methods: Class Lecture

**Chapter 20**

Class hours: 10 hours

Main Content: Styling and Deploying an App

Teaching requirements:**style the entries on the topic page…443, deleting a project on platform.sh..461**

Key points: give projects a simple but professional appearance using the Bootstrap library and the django-bootstrap5 app, Bootstrap templates, use the Navbar static template to create Learning Log, use a jumbo-tron, style all pages in a site consistently, deploy a project to a remote server so anyone can access it, make a Platform.sh account and install some tools that help manage deployment process, use Git to commit working project to repository, push repository to a remote server on Platform.sh, begin securing your app by by setting DEBUG = False on the live server, make custom error pages so errors that arise will look well handled.

Difficult points: **Exercise 20-2: Stylish Blog, Exercise 20-4: Extended Learning Log**

Teaching methods: Class Lecture

Exam Review

Class hours: 2 hours

Main Content: Review of course contents

Teaching requirements:

Key points: N/A

Difficult points: N/A

Teaching methods: Class Lecture

**2.Practical (experimental, training, other) teaching**

（1） Practical projects and class hours

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Basis of Project | Project Content (Experiment, Practical Training, Other) | **Class hours** | Remark |
| 1 | Ch 15: Generating Data | 15-10. Practicing with Both Libraries: Try using Matplotlib to make a die-rolling  visualization, and use Plotly to make the visualization for a random walk. (You’ll  need to consult the documentation for each library to complete this exercise.) | 1 hour in class + hw hours | Show flash screens of your work and describing it.  Working in groups is ok.  Use group resources or lab to create a physical network. If you can’t then detail how the project that requires network pc’s would have worked.  If these projects are too hard, the student can change it to any other in TRY IT YOURSELF. More points will be rewarded for the harder project. |
| 2 | Ch 16: Downloading Data | 16-9. World Fires: In the resources for this chapter, you’ll find a file called  *world\_fires\_1\_day.csv*. This file contains information about fires burning in differ  ent locations around the globe, including the latitude, longitude, and brightness  of each fire. Using the data-processing work from the first part of this chapter  and the mapping work from this section, make a map that shows which parts of  the world are affected by fires.  You can download more recent versions of this data at *https://earthdata*  *.nasa.gov/earth-observation-data/near-real-time/firms/active-fire-data*. You can  find links to the data in CSV format in the *SHP, KML, and TXT Files* section. | 1 hour in class + hw hours |
| 3 | Ch 17: Working with APIs | 17-4. Further Exploration: Visit the documentation for Plotly and either the GitHub API or the Hacker News API. Use some of the information you find there to either customize the style of the plots we’ve already made or pull some different information and create your own visualizations. If you’re curious about exploring other APIs, take a look at the APIs mentioned in the GitHub repository at *https://github.com/public-apis*. | 1 hour in class + hw hours |
| 4 | Ch 18: Getting Started with Django | 18-8. Pizzeria Pages: Add a page to the Pizzeria project from Exercise 18-6 (page 392) that shows the names of available pizzas. Then link each pizza name to a page displaying the pizza’s toppings. Make sure you use template inheritance to build your pages efficiently. | 1 hour in class + hw hours |
| Total | **4 hours in-class collaboration,** unlimited homework hours for the project. | **Projects are to be reviewed a total 1 hour and unlimited hours as homework to finish the project. Projects are due latest after completion of chapter.** | 4 hours in class + hw hours |

(2) Basic teaching requirements

**Project 1**

Project name：Generating Data

Teaching purpose: How to consult documentation on a library to complete a project. **1.**Library familiarity; encourages students to explore and become familiar with both Matplotlib and Plotly libraries. Helps them understand the strengths and weaknesses of each library for different types of visualizations. **2.**Documentation Usage; teaches students how to effectively use library documentation to find the necessary functions and methods to implement specific visualizations, enhance their ability to learn independently and utilize external resources for problem-solving. **3.**Data Visualization Concepts; reinforces fundamental concepts of data visualization such as representing data, exploring distributions, and conveying patterns or trends, helps students understand how to choose appropriate visualization techniques for different types of data and analysis tasks. **4.**Customizing and Styling; encourages students to experiment with customization options and styling features offered by both libraries, promotes creativity in designing visually appealing and informative plots. **5.**Comparative Analysis; provides an opportunity for students to compare and contrast the capabilities and syntax of Matplotlib and Plotly, helps them make informed decisions about which library to use for different projects based on factors like ease, interactivity, and performance.

Basic requirement：Ch15 p312 Matplotlib die rolling visualization; create a visualization that simulates rolling a die, display the outcome of each roll, show the distribution of outcomes over multiple rolls (histogram or bar plot). Ch15 p319 Plotly random walk visualization; implement a random walk algorithm, visualize the random walk using plotly, include features like trace lines, markers, and annotations, customize the plot with appropriate labels, titles, and styling.

**Project 2**

Project name：Downloading Data

Teaching purpose： **1.** Data Retrieval and Management; understanding how to access and download real-world datasets from reliable sources like NASA’s Earth Observation Data portal. **2.** Data processing; reinforcing skills in data manipulation and extraction from CSV files, including handling missing values and formatting issues. **3.** Geospatial Visualization; introducing geospatial mapping concepts and techniques for visualizing data on geographical maps using latitude and longitude coordinates. **4.**Data Interpretation; encouraging students to analyze and interpret geographical data to identify patterns, trends, and anomalies, such as regions with high fire activity. **5.**Problem-solving; fostering problem-solving skills by providing a practical project that requires students to apply their programming and data analysis knowledge to address a real-world issue. **6.**Documentation and Communication; emphasizing the importance of clear documentation and effective communication in conveying the steps taken and the insights gained during the data analysis process.

Basic requirement: **1.** Data Downloading; students need to download the dataset “world\_fires\_1\_day.csv” from the provided link or any more recent versions available at the given NASA website. **2.** Data Processing; utilize the techniques learned in Chapter 16 for processing csv files, extracting relevant information such as latitude, longitude, and brightness of each fire. **3.** Mapping; apply mapping techniques covered in the chapter to create a visualization that depicts the locations affected by fires on a world map. **4.**  Visualization; the map should effectively represent the distribution of fires across the globe, with different markers or color intensities indicating the intensity or frequency of fires in specific regions. **5.**Documentation; provide clear documentation of the steps taken, including data preprocessing, mapping techniques employed, and any additional features added to enhance the visualization. **6.**Code; develop python code (or any other relevant programming language) to automate the data processing and mapping tasks. *Suggested reading Ch16 p342*

**Project 3**

Project name： Working with APIs

Teaching purpose： **1.**Understanding APIs; gain familiarity with the concept of APIs and how they allow interaction between different software applications. **2.**API Documentation and Interpretation; learn how to read and interpret API documentation provided by various services, such as Plotly and Github/Hacker News. **3.**Data Visualization Customization; practice customizing data visualizations using external APIs, enhancing the aesthetic appeal and functionality of plots. **4.**Data Retrieval and Manipulation; gain experience in retrieving data from APIs, manipulating it as needed, and integrating it into data visualizations. **5.**Exploring External APIs; encourage exploration of different APIs beyond those covered in class, fostering curiosity and independent learning.

Basic requirement: **1.**API Selection; choose either the Github API or the Hacker News API for data retrieval and visualization. Optionally, explore additional APIs listed in the github repo. **2.**Documentation Review; Visit the documentation for Plotly as well as the chosen API to understand their functionalities and usage. **3.**Data Visualization Enhancement; customize the style of existing plots created in previous exercises using Plotly. Experiment with different layout options, color schemes, and annotations to improve visual clarity and aesthetics. **4.**Data Retrieval and Visualization; retrieve relevant data from the chosen API and create new visualizations based on it. This could involve pulling different types of information, such as github repo’ stats or trending topics on Hacker News, and representing them graphically. **5.**Code Implementation; write code to interact with both the Plotly library for visualization and the chosen API for data retrieval. Ensure that the code is well-documented and followsbest practices for readability and maintainability. **6.**Presentation or Documentation; prepare a presentation or documentation summarizing the project, including the chosen API, the approach taken for data visualization customization, and the insights gained from exploring external APIs.

*Suggested reading Ch17 p356*

**Project 4**

Project name： Getting Started with Django

Teaching purpose： **1.**Template Inheritance; understand and implement the concept of template inheritance to efficiently manage and reuse code across multiple pages in a web application. **2.**Dynamic Content; Learn how to dynamically generate content based on user input or database values to create dynamic web pages. **3.**Routing and Linking; practice creating routes and linking pages within a web application to ensure seamless navigation for users. **4.**Data Management; explore techniques for managing and displaying data, such as storing pizza names and toppings, and retrieving them to populate web pages. **5.**User Experience; focus on enhancing the user experience by organizing information in a clear and intuitive manner, allowing users to easily find and navigate through available pizza options.

**Basic requirement:**  **1.**Add a Page for Available Pizzas; create a new page within the Pizzeria project to display the names of available pizzas. This page should be linked to the existing project from Exercise 18-6 or your chosen project. **2.**Link Pizza Names to Toppings; ensure that each pizza name displayed on the page is clickable and linked to a separate page displaying the toppings of that particular pizza. **3.**Template Inheritance; utilize template inheritance to efficiently build the new pages, inheriting common layout elements from a base template to maintain consistency across the application. **4.**Dynamic Content Generation; dynamically generate the content for both the pizza names page and the individual pizza toppings pages based on data stored in the application, such as pizza names and their corresponding toppings. **5.**Navigation; implement intuitive navigation features, such as breadcrumbs or a navigation bar, to allow users to easily move between pages within the application. **6.**Testing and Validation; ensure that the implemented features are thoroughly tested to verify their functionality and address any potential issues or bugs. Validate user inputs and handle edge cases appropriately.

*Suggested reading Ch18 p375*

**Ⅲ、Course chapters and class hour allocation table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| Chapter | Teaching content | Page count | Teaching theory | Experimental Hours | Practical training hours | Total |
|  | **PART I: BASICS** |  | Lecture | Classwork/Lab | Projects (are part of hw) |  |
| 1 | Getting Started | 3-15 | 1.5 | 0.5 |  | 2 |
| 2 | Variables and Simple Data Types | 15-33 | 1.5 | 0.5 |  | 2 |
| 3 | Introducing Lists | 33-49 | 1.5 | 0.5 |  | 2 |
| 4 | Working with Lists | 49-71 | 1.5 | 0.5 |  | 2 |
| 5 | if Statements | 71-91 | 1.5 | 0.5 |  | 2 |
| 6 | Dictionaries | 91-113 | 1.5 | 0.5 |  | 2 |
| 7 | User Input and while Loops | 113-129 | 1.5 | 0.5 |  | 2 |
| 8 | Functions | 129-157 | 1.5 | 0.5 |  | 2 |
| 9 | Classes | 157-183 | 1.5 | 0.5 |  | 2 |
| 10 | Files and Exceptions | 183-209 | 1.5 | 0.5 |  | 2 |
| 11 | Testing your Code | 209-225 | 1.5 | 0.5 |  | 2 |
|  | **PART II: PROJECTS** |  |  |  |  | 2 |
| 12 | A Ship That Fires Bullets | 225-255 | 1.5 | 0.5 |  | 2 |
| 13 | Aliens! | 255-277 | 1.5 | 0.5 |  | 2 |
| 14 | Scoring | 277-301 | 1.5 | 0.5 |  | 2 |
| 15 | Generating Data | 301-329 | 1.5 | 0.5 |  | 2 |
| 16 | Downloading Data | 329-355 | 1.5 | 0.5 |  | 2 |
| 17 | Working with APIs | 355-373 | 1.5 | 0.5 | 1 | 3 |
| 18 | Getting Started with Django | 373-403 | 1.5 | 0.5 | 1 | 3 |
| 19 | User Accounts | 403-433 | 1.5 | 0.5 | 1 | 3 |
| 20 | Styling and Deploying an App | 433-463 | 1.5 | 0.5 | 1 | 3 |
|  | File Handling |  | 1 | 0.0 |  | 1 |
|  | Course Review |  | 2 | 0 |  | 4 |
|  |  |  | 34 | 10 | 4 | 48 |

**Ⅳ、Examination and evaluation structure ratio**

1. **Examination method**

The final exam for this course is a hand written exam. Students will be allowed to bring in a study sheet (1 A4 sized paper). The exam may be taken as writing code on paper. The student should be prepared for this.

**2.Composition of test scores**

The final score for this course will include a 40% final exam, and 4 assignments and 2 in class assessments that makes up 60% (10%/assignment,assessment).

**Ⅴ、Proposition requirements**

**Ⅵ、Course materials and main reference books**

**1.Required textbook**

Python Crash Course, 3rd edition: Matthes, E.

1. **Recommended reference books**

Cracking the Coding Interview: McDowell, G.

Introduction to Algorithms: Cormen, T.

Data Structures and Algorithms in Python: Goodrich, M.

**Ⅶ、Compiling Explanation**

**1.Compilation basis**

**2.Connection and coordination with relevant courses and content**

N/A

This is an introductory course and therefore has no prerequisite, however, familiarity with any unix type system, and any programming language, is useful.

**3.Others**

Course Syllabus written by：Michael Smith

Course Syllabus reviewed by：Long Rui

Revision Date：2024.4.1