2025

LEARNER GUIDE 2025

OCC: Software Developer
NQF 5
COMPUTING THEORY
COT512

2025

CTU training solutions | Creative & Technology Universitas

Table of Contents

Occupational Certificate:

Software Developer

(NQF 5)
(SAQA ID: 118707)

LEVEL	SAQA ID
NQF 5	118707

How to use the guide

The guide will provide an overview off the syllabus and will provide the learning outcomes of the module. It will indicate each major topic that will be covered, as well as the learning outcomes of each topic.
The study guide is NOT a replacement of textbooks and should be studied in conjunction with the required textbooks.
At the end of each study unit there will be a summary, followed by several self-assessment questions. These questions will assist you to prepare for the tests and exams.

The following icons will be used in the study guide:

	Sections in the prescribed textbook that the student needs to study
	Additional reading that the student needs to study
	A video that the student needs to watch
	Class activities to be completed
	Activities to be uploaded to CampusOnline
	Exercises to be completed
	Indicates exercises to be uploaded to CampusOnline
	Flipped Classroom

	Group activities to be completed
	Group activities to be uploaded to CampusOnline
	Relevant points for the student to consider
	Projects to be completed
	Tests to be completed
	Revision questions to be completed
	Webinar to be attended
	A field trip to be attended

Introduction

This Learner Guide provides a comprehensive overview of the module. It is designed to improve the skills and knowledge of learners and thus enabling them to effectively and efficiently complete specific tasks.

Purpose of module

The main focus of the learning in this knowledge module is to build an understanding of programming as creating a set of instructions to a computer on how to perform a task using coding and programming languages as well as providing the learner with an opportunity to acquire the ability to apply basic programming skills and code to use a software toolkit/platform in the field of study.

Duration

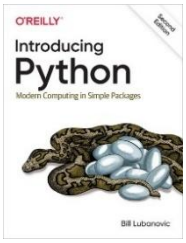
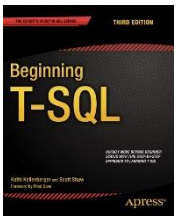
The total notional hours will be allocated according to the table below:

Proposed Roll Out Strategy	25/08/2025 - 23/09/2025
Credits	5
Total Notional Hours:	50
Theory	15
Practical	11
Contact Sessions	17
Formative Assessments	2
Summative Assessments	3

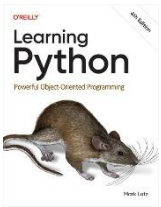
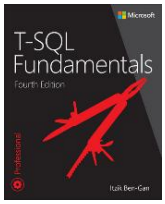
Learning outcomes / Unit standards

- KM-05-KT01: Introduction to programming languages
- KM-05-KT02: Programming basics
- KM-05-KT03: Software applications
- PM-07-PS01: Source and compare at least three software toolkits/platforms/ languages used in your field of studies
- PM-07-PS02: Identify and contrast four (4) paradigms
- PM-07-PS03: Create a programming environment (tailored to a specific tool or platform)
- PM-07-PS04: Write code using a programming language for giving instructions for use of a software toolkit/platform
- PM-07-PS05: Select and use correct data types (tailored to a specific tool or platform)
- PM-07-PS06: Use complex types to organise data (tailored to a specific tool or platform)
- PM-07-PS07: Add API's (Application Programming Interface) to an application (tailored to a specific tool or platform)
- PM-07-PS08: Define a function (tailored to a specific tool or platform)
- PM-07-PS09: Use logical branch statements and comparison operators (tailored to a specific tool or platform)
- PM-07-PS10: Code loops (tailored to a specific tool or platform)
- PM-07-PS11: Use and apply variable scopes (tailored to a specific tool or platform)
- PM-07-PS12: Create queries to pull desired data using a structured query language (SQL) (applicable to data base) (tailored to a specific tool or platform)
- PM-07-PS13: Handle errors and troubleshooting (tailored to a specific tool or platform)
- PM-07-PS14: Identify the general steps for writing code (tailored to a specific tool or platform)

Module Resources

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/	
Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/	

Recommended additional reading

Learning Python by Mark Lutz. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/learning-python-6th/9781098171292/	
T-SQL Fundamentals by Pearson Education, Inc. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/t-sql-fundamentals-4th/9780138101930/	

Formative assessment breakdown

Formative 1	Formative 2
Project 1	Project 2
50%	50%

Summative assessment:

Practical Project	Exam
50%	50%

Formative assessments (50%) + **Summative** assessment (50%) = **Final mark**

Practical assessment breakdown

Practical (PM-07)
Project
100%

Assessment Preparation Guidelines

Formative assessment 1:	**Format of the Assessment:** Students will receive a research project that will require them to study a scenario and develop a solution in the form of an application. **Learning outcomes covered:** • Introduction to programming languages • Programming basics **Resources required:** • MS Word • VS Code • Python
Formative assessment 2:	**Format of the Assessment:** Students will receive a research project that will require them to study a scenario and develop a solution in the form of an application. **Learning outcomes covered:** • Introduction to programming languages • Programming basics • Software applications **Resources required:** • MS Word • VS Code • Python
Exam:	**Format of the Assessment:** Students will write a written theory exam on campus under invigilation. This will be a closed book exam. Students will also receive a project that will count 50% of the Summative mark. **Learning outcomes covered:** • Introduction to programming languages • Programming basics • Software applications **Resources required:** • MS Word • VS Code • Python • SQL Server • SSMS

Practical Preparation Guidelines

Practical (PM-07):

Format of the Assessment:

Students will receive a research project that will require them to study a scenario and develop a solution in the form of an application.

Learning outcomes covered:

- Source and compare at least three software toolkits/platforms/ languages used in your field of studies
- Identify and contrast four (4) paradigms
- Create a programming environment (tailored to a specific tool or platform)
- Write code using a programming language for giving instructions for use of a software toolkit/platform
- Select and use correct data types (tailored to a specific tool or platform)
- Use complex types to organise data (tailored to a specific tool or platform)
- Add API's (Application Programming Interface) to an application (tailored to a specific tool or platform)
- Define a function (tailored to a specific tool or platform)
- Use logical branch statements and comparison operators (tailored to a specific tool or platform)
- Code loops (tailored to a specific tool or platform)
- Use and apply variable scopes (tailored to a specific tool or platform)
- Create queries to pull desired data using a structured query language (SQL) (applicable to data base) (tailored to a specific tool or platform)
- Handle errors and troubleshooting (tailored to a specific tool or platform)
- Identify the general steps for writing code (tailored to a specific tool or platform)

Resources required:

- MS Word
- VS Code
- Python
- SQL Server
- SSMS

Formative Assessment Submissions:

Formative assessment:	Assessment description:	Submission:
FA 1	Research (Project 1)	Week 2
FA 2	Research (Project 2)	Week 3
Practical (PM-07)	Project	Week 4

Please note – There are two (2) steps in the submission process.

- Step 1: Required evidence in the specified formats are submitted on Campus Online to the designated assignment description.
- Step 2: Complete and submit document of authenticity for every formative and summative assessment submitted.

Weekly planner 2025

Campus Week	Module Week	Dates	Holidays and exams
Week 1			
Week 2			
Week 3			
Week 4			
Week 5			
Week 6	Week 1	25 Aug – 29 Aug	
Week 7	Week 2	01 Sept – 05 Sept	
Week 8	Week 3	08 Sept – 12 Sept	10 Sept – 12 Sept Sports Day
Week 9	Week 4	15 Sept – 19 Sept	Summative Practical
Week 10	Week 5	22 Sept – 26 Sept	Summative Theory & Internationals
Week 11	Week 1		New Module Starts
Week 12	Week 2		
Week 13	Week 3		
Week 14	Week 4		
Week 15	Week 5		
Week 16	Week 6		
Week 17			
Week 18			
December Holidays			

Lesson Plan:

Week 1:

Theme:

Introduction to programming languages

- Textbook 1
  - Chapter 1

Programming basics

- Textbook 1
  - Chapter 2 - 6

Class activity:

Facilitator to demonstrate practical steps where applicable.

Material requirements:

- Learner Guide
- Textbook 1: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Week 2:

Theme:

Programming basics

- Textbook 1
  - Chapter 7 - 9

Software applications

- Textbook 1
  - Chapter 10 - 15

Class activity:

Facilitator to demonstrate practical steps where applicable.

Material requirements:

- Learner Guide
- Textbook 1: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Week 3:

Theme:

Software applications

- Textbook 1
  - Chapter 15 – 16
- Textbook 2
  - Chapter 1 - 3

Class activity:

Facilitator to demonstrate practical steps where applicable.

Material requirements:

- Learner Guide
- Textbook 1: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/
- Textbook 2: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

<table>
<tr>
<td>Week 4:</td>
<td>
Theme:
Software applications
• Textbook 1
○ Chapter 17 - 22

Class activity:
Facilitator to demonstrate practical steps where applicable.

Material requirements:
• Learner Guide
• Textbook 1: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/
</td>
</tr>
<tr>
<td>Week 5:</td>
<td>
• Summative Exams
• International Exams (ITS- Python)
</td>
</tr>
</table>

Week 1

Day 1 – Learning Unit 1

Introduction

This module will introduce students to the basics of programming, including concepts, principles and terminology. This chapter also covers some of the different data types in Python. We will also cover working with numbers in Python. The facilitator will cover slide 1 and is linked to Textbook 1, chapters 1 to 3.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Introduction to programming languages

Additional Resources

Concepts, principles and terminology is described

The Principle of Programming Language comes from concatenating of Principle + Programming Languages. By defining principles, it is the fundamental norms, values, rules and regulations that are uses to represent what is desirable and positive for any job or task, it is helpful in determining the rightfulness or wrongfulness of any state task. In formal principles are very basic than policy and objectives, and are meant to govern both. While programming languages relates to the formal construction of languages by expert programmers that leads to communicating high level instruction (e.g. human instructions) to a machine (e.g. General Computer) or a specific computer (e.g. Application Based Computers). A programming language can be used to create application programs, scripts or other sets of instructions to control the behaviour of a machine or to express procedural steps called algorithms.

Hence, we can define the principle of programming language as, "Principle of Programming Language is a set of rules and norms governed to communicate instructions (high level instruction or assemble level instruction) to a machine or particularly a computer."

The concepts of General-purpose computers have the amazing huge collection of instruction set that are useful to coordinate human desirable instruction to machine understandable. On the basis of set of instructions called programs, a general-purpose computer becomes a single piece of hardware that can able to do any computation imaginable. To handle the action of general-purpose computer, programmers uses the specific programming language based on those prefix set of instructions by programming language programmers develop computational code that handle computer actions.

What are the basic fundamental concepts of programming?

Irrespective of the programming language you choose to learn, the basic concepts of programming are similar across languages. Some of these concepts include:

- Variable Declaration
- Basic Syntax
- Data Type and Structures
- Flow Control Structures (Conditionals and loops)
- Functional Programming
- Object-Oriented Programming
- Debugging
- IDEs and Coding Environments

In the next section of this shot, you will be given a brief introduction to these concepts.

Variable declaration
Variables are containers for storing data values, a memory location for a data type. Variables are created using a declaration or keyword that varies across languages.

Variable names are usually alphanumeric, that is, they contain a-z and 0-9. They can also include special characters like underscore or the dollar sign.

Variables can hold values of any data type supported by the programming language. This value may change during program execution.

Basic syntax

Every programming language has its syntax, and you must learn the fundamental syntax of the language you are learning.

Syntax refers to the set of rules that define the structure of a language. It is almost impossible to read or understand a programming language without its syntax.

Revision Questions Day 1

1. What does the command import webbrowser do?
    A. It imports a module that allows the program to open web pages in a browser.
    B. It opens a new web browser window.
    C. It imports a web browser into the Python program.
    D. It imports the ability to manipulate web browser settings.

2. What is the purpose of the import statement?
    A. To print text to the display
    B. To connect to a web server and request a particular web service
    C. To read what the user types and save it in a variable
    D. To make available to the program all the code from a specific Python standard library module

3. Which of the following is not a reason for Python's popularity?
    A. It's the fastest-growing major programming language.
    B. It's the most popular language for introductory computer science courses at top American colleges.
    C. It's the official teaching language for high schools in France.
    D. It's the only programming language that can be used for data science and machine learning.

4. In Python, which of the following variable names is considered a good balance between brevity and clarity?
    A. gaviidae_inventory
    B. num_loons
    C. a
    D. x

5. In Python, which of the following is not a valid variable name?
    A. a1
    B. another-name
    C. a
    D. a_b_c___95

6. In programming, what does the "=" symbol represent?
    A. Equality
    B. Calculation
    C. Initialization
    D. Assignment

7. In Python, which of the following data types is mutable?
    A. str
    B. tuple
    C. list
    D. int

8. In Python, what is the prefix used to specify a binary integer?
    A. 0b
    B. 0x
    C. 0o
    D. 0d

9. In Python, what happens when you mix integers and floats in an operation?
    A. The float values are automatically demoted to integer values.
    B. The integer values are automatically promoted to float values.
    C. An error is thrown.
    D. The operation is not allowed.

10. In Python, what is the result of converting a boolean True to a float?
    A. None
    B. 1.0
    C. 0.0
    D. Error

Class activity 1

Class Activity: Starting with Python
Objective:

To help students become familiar with the basics of Python.

Task

Complete the following tasks in VS Code using Python

1. Assign the integer value 99 to the variable prince and print it.
2. What type is the value 2.0?
3. What type is the expression 5 + 2.0?
4. How many seconds are there in an hour? Use the interactive interpreter as a calculator and multiply the number of seconds in a minute (60) by the number of minutes in an hour (also 60).
5. Assign the result from the previous task (seconds in an hour) to a variable called seconds_per_hour.
6. How many seconds are there in a day? Use your seconds_per_hour variable.
7. Calculate seconds per day again, but this time save the result in a variable called seconds_per_day.
8. Divide seconds_per_day by seconds_per_hour. Use floating-point (/) division.
9. Divide seconds_per_day by seconds_per_hour, using integer (//) division. Did this number agree with the floating-point value from the previous question, aside from the final .0?

Homework: Facilitator to give additional homework activity

Formative 1 Project

FORMATIVE 1 – (PROJECT 1) TO BE HANDED OUT TO STUDENTS

Project 1

Read through the briefing document with the students and emphasize evidence requirements in the project including submission format.

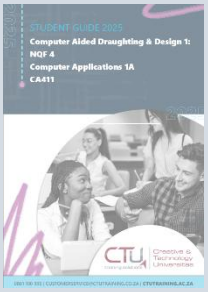
Brief description

Students will receive a research project that will require them to study a scenario and develop a solution in the form of an application.

Learning outcomes

- **Introduction to programming languages**
- **Programming basics**

Formative 2 Project

FORMATIVE 2 – (PROJECT 2) TO BE HANDED OUT TO STUDENTS	

Project 2

Read through the briefing document with the students and emphasize evidence requirements in the project including submission format.

Brief description

Students will receive a research project that will require them to study a scenario and develop a solution in the form of an application.

Learning outcomes

- Introduction to programming languages
- Programming basics
- Software applications

PM-01 Project

PRACTICAL PROJECT – PM-07 TO BE HANDED OUT TO STUDENTS

Project

Read through the briefing document with the students and emphasize evidence requirements in the project including submission format.

Brief description

Students will receive a research project that will require them to study a scenario and develop a solution in the form of an application.

Learning outcomes

- Source and compare at least three software toolkits/platforms/ languages used in your field of studies
- Identify and contrast four (4) paradigms
- Create a programming environment (tailored to a specific tool or platform)
- Write code using a programming language for giving instructions for use of a software toolkit/platform
- Select and use correct data types (tailored to a specific tool or platform)
- Use complex types to organise data (tailored to a specific tool or platform)
- Add API's (Application Programming Interface) to an application (tailored to a specific tool or platform)
- Define a function (tailored to a specific tool or platform)
- Use logical branch statements and comparison operators (tailored to a specific tool or platform)
- Code loops (tailored to a specific tool or platform)
- Use and apply variable scopes (tailored to a specific tool or platform)
- Create queries to pull desired data using a structured query language (SQL) (applicable to data base) (tailored to a specific tool or platform)
- Handle errors and troubleshooting (tailored to a specific tool or platform)
- Identify the general steps for writing code (tailored to a specific tool or platform)

Day 2 - Learning Unit 2

Introduction

In this chapter, we look at comments, escape sequences. Students will also be introduced to if statements and how to make logical decisions using these statements in applications. The facilitator will cover slide 2 and is linked to Textbook 1 chapter 4.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Programming basics

Additional Resources

An understanding of the basics of a programming language is demonstrated

What Is A Programming Environment?

If you're not well-versed in coding, the phrase "programming environment" may sound foreign to you. However, when leading or participating in a project designed to deliver any software component, understanding the concept is necessary to ensure success.

In a general sense, a programming environment combines hardware and software that allows a developer to build applications.

Developers typically work in integrated development environments or IDEs. These connect users with all the features necessary to write and test their code correctly. Different IDEs will offer other capabilities and advantages.

This article will outline the specifics of an IDE and detail the nuances of some potential programming environments your business might employ when developing an app.

What Is An IDE?

IDEs afford developers a convenient workspace when embarking on a project by packaging all the development tools a developer needs into a single graphical user interface.
They typically provide a space for source code editing and a debugging program that locates problems in any code that has been written. Additionally, IDEs offer automation functionality that can handle complaining, packaging, and testing code.
The popularity of IDEs in the development community hinges on their convenience. While integrating components can offer a fuller app customization experience, using an IDE provides a streamlined development process while minimizing the potential for user error.
While IDEs are instrumental to the development process, finding the right one for your project can be difficult. Questions such as the following may come up in your IDE selection process:

- What language will my software development team use to develop the app? Does it need to be compatible with any specific platforms or operating systems?
- Do we anticipate needing to add additional functionality to our IDE to handle project requirements?
- Which parts of the development process will this IDE allow me to automate?
- Will the devices we use to build the solution be able to handle the amount of memory used by the IDE?

With no shortage of IDE options available, understanding their general capabilities and attributes is key to making the right programming environment decision for your project.
Examples Of Programming Environments
Being familiar with popular programming environment options will help you make an informed decision for your app-building endeavours. Four of the most common IDEs are IntelliJ, Eclipse, NetBeans, and Visual Studio.
Visual Studio
Visual Studio is an IDE developed by Microsoft. It offers a great deal of versatility, providing users with an expansive library of extensions that allows for more customization than other environments. Compatibility testing in Visual Studio is also difficult to match.

Revision Questions Day 2

1. In Python 3.8, what is the purpose of the walrus operator :=?
    A. To assign a value to a variable
    B. To check if two values are equal
    C. To check if a value is part of a sequence
    D. To assign a value to a variable as part of an expression

2. In Python, what is the result of the following comparison: (5 < x) and (x < 10), if x = 7?
    A. None
    B. True
    C. False
    D. Error

3. In Python, which of the following is considered False?
    A. Nonempty dictionary
    B. Nonzero integer
    C. Nonempty string
    D. Empty list

4. In Python, what does the membership operator in do?
    A. Checks if a value is greater than another
    B. Checks if a value is less than another
    C. Checks if a value is equal to another
    D. Checks if a value is part of a sequence

5. In Python, what is the recommended maximum line length?
    A. 70 characters
    B. 80 characters
    C. 60 characters
    D. 90 characters

6. What is the purpose of the backslash \ in Python code?
    A. It begins a comment
    B. It allows for line continuation
    C. It escapes special characters only
    D. It ends a function definition

7. What happens if you forget the colon : at the end of an if statement in Python?
    A. Python treats it as a comment
    B. It will still run, but with a warning
    C. Python will raise a syntax error
    D. It will automatically insert a colon

8. What is the primary way Python defines the structure of code blocks?
    A. Using curly braces {}
    B. By declaring begin and end
    C. Through consistent indentation
    D. With semicolons at the end of lines

9. What character is used to create a comment in Python?
    A. //
    B. <!—
    C. /* */
    D. #

10. What does the in keyword do in the expression letter in 'aeiou'?
    A. Compares memory addresses
    B. Checks if letter exists in the sequence 'aeiou'
    C. Converts letter to uppercase
    D. Splits the string into a list

Class activity 2

Title: "Choosing with if"
Objective:

Students will create multiple small projects to illustrate their ability to apply if statements to perform logical decisions using Python.

Instructions:

Complete the following tasks using Python and VSCode.

1. **Task 1**
   - Prompt the user to enter a number and using an if statement, determine if the user guessed the correct number, printing "Congratulations you guessed correctly" if the user guessed the correct number.

2. **Task 2**
   - Use the code from the previous task and append the code to display a message to the user if the user guessed the incorrect number.

3. **Task 3**
   - Use the code from the previous task and determine if the user guessed a number close to the number and print a message to the user "You were so close".

4. **Task 4**
   - Prompt the user to enter a color and based on the color display a message as follows:
     - Yellow – "The sun is yellow"
     - Green – "The grass is green"
     - Blue – "The ocean is blue"
     - Brown – "The ground is brown"
     - Any other color – "Sorry no such color in the registry"

Homework: Facilitator to give additional homework activity

Day 3 - Learning Unit 3

Introduction

In this chapter, we will look at strings and string operations. Students will learn how to create strings with str(), as well as duplicating strings, extracting characters from a string and other string operations. The facilitator will cover slide 3 and is linked to textbook 1 chapter 5.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Programming basics

Additional Resources

Common string operations and methods.
String Creation and Basics:
Defining Strings.

```
my_string = "Hello, World!"
another_string = 'Python is fun.'
multi_line_string = """This is a
                        multi-line string."""     #Referred to as a doc-string
```

Immutability: Strings in Python are immutable, meaning their content cannot be changed after creation. Operations like replace() return a new string.

Indexing and Slicing:
Accessing Characters.

```
s = "Python"
print(s[0])  # Output: P
print(s[-1]) # Output: n (last character)
```

slicing.

```
s = "Programming"
print(s[0:3])   # Output: Pro (from index 0 up to, but not including, 3)
print(s[4:])    # Output: ramming (from index 4 to the end)
print(s[:3])    # Output: Pro (from the beginning up to, but not including, 3)
print(s[::2])   # Output: Prgamn (every second character)
print(s[::-1])  # Output: gnimmargorP (reverse string)
```

String Methods:
Case Conversion:

- s.lower(): Returns a lowercase copy.
- s.upper(): Returns an uppercase copy.
- s.capitalize(): Capitalizes the first character.
- s.title(): Capitalizes the first character of each word.
- s.swapcase(): Swaps uppercase to lowercase and vice versa.

Whitespace Manipulation:

- s.strip(): Removes leading/trailing whitespace.
- s.lstrip(): Removes leading whitespace.
- s.rstrip(): Removes trailing whitespace.

Searching and Replacing:

- s.find(substring): Returns the lowest index of the substring, or -1 if not found.
- s.index(substring): Similar to find(), but raises a ValueError if not found.
- s.count(substring): Returns the number of occurrences of the substring.
- s.replace(old, new): Returns a new string with old replaced by new.

Splitting and Joining:

- s.split(delimiter): Returns a list of substrings split by the delimiter (defaults to whitespace).
- delimiter.join(list_of_strings): Joins elements of an iterable with the delimiter.

Content Checking (returns True/False):

- s.isalpha(): All characters are alphabetic.
- s.isdigit(): All characters are digits.
- s.isalnum(): All characters are alphanumeric.
- s.isspace(): All characters are whitespace.
- s.startswith(prefix): Starts with the given prefix.
- s.endswith(suffix): Ends with the given suffix.

String Formatting:
f-strings (Formatted String Literals).

```
name = "Alice"
age = 30
print(f"Name: {name}, Age: {age}")
```

.format() method.

```
print("Name: {}, Age: {}".format(name, age))
```

Old-style % formatting.

```
print("Name: %s, Age: %d" % (name, age))
```

Concatenation:

+ operator: s1 + s2 concatenates strings.
* operator: s * n repeats the string n times.

Class activity 3

Title: "Text Strings"

Objective:

Students will need to perform several operations on text strings.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

Use the following text to complete the tasks below. The poem by Alan Bender

The strange quantum uncertainty
here, there, and everywhere,
within, without, nonrandom bits sustain
universal poetic harmony.
The aetheric history matter dark intones

mystically to a poet on a quantum trip
entangled with monkeys
programming an escape from reality.
The centaur got a gyre wiring diagram
Took the labyrinth to MIT

Yeats was rejected by editors
for anticipating Asimov.
Einstein's E got lost forever
in an alternate frame of reference.
And,42 was the answer we seek after all.

"was" that nonlocal past tense rheomode
phenomenon of the explicate order.
Mind - matter measurement scale factors
aside, consciousness, the muse,
the psi of Pythagorean dream, endures.

1. **Task 1**
    - Assign the poem to a variable named poem
    - Print out the poem to the user.

2. **Task 2**
    - Use the proper escape sequences to make the poem more readable to the user

3. **Task 3**
    - Extract the text from the word “Mind” to the end of the poem and print it

4. **Task 4**
    - Determine the length of the poem string and print it out to the user

5. **Task 5**
    - Use the appropriate string function to split the poem on every new line
    - Store this in a variable called poem_lst
    - Print the list to the user

6. **Task 6**
    - Use an appropriate function to determine the index position of the word “labyrinth” and print it to the user

7. **Task 7**
    - Use the appropriate text function to count the occurrences of the word “The”.

Homework: Facilitator to give additional homework activity

Revision Questions Day 3

1. In Python, which method is used to remove leading or trailing characters from a string?
    A. erase()
    B. remove()
    C. strip()
    D. delete()

2. In Python, which string formatting style is recommended as of Python 3.6?
    A. Old-style formatting
    B. No specific style is recommended
    C. New-style formatting
    D. f-strings

3. In Python, which method is used to convert all characters in a string to uppercase?
    A. capitalize()
    B. upper()
    C. toUpper()
    D. upperCase()

4. In Python's new-style formatting, what does the alignment character ^ signify?
    A. Center alignment
    B. No alignment
    C. Left alignment
    D. Right alignment

5. In Python, what does the escape sequence \n represent?
    A. A new tab
    B. A new paragraph
    C. A new space
    D. A new line

6. True or false: In Python, you can directly insert a character into a string or change the character at a specific index.
    A. True
    B. False

7. In Python, which function would you use to break a string into a list of smaller strings based on some separator?
    A. len()
    B. split()
    C. replace()
    D. join()

8. True or false: Strings in Python are immutable.
    A. True
    B. False

9. Which of the following is the correct way to extract a character from a string of characters?
    A. letter = word(2)
    B. letter = word{2}
    C. letter = word[2]
    D. letter = word:2

10. What datatype would be produced by doing the following:
    tasks = 'get gloves,get mask,give cat vitamins,call ambulance'
    task = tasks.split(',')
        A. List
        B. Dictionary
        C. Set
        D. Tuple

Day 4 – Learning Unit 4

Introduction

This chapter introduces students to Python loops. Students will learn how to loop through a block of code using while and for loops. The facilitator will cover slide 4 and is linked to textbook 1 chapter 6.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Programming basics

Additional Resources

Common loop constructs and control flow statements.

For Loops:
Iterating over a sequence (list, tuple, string, etc.):

```
my_list = [1, 2, 3]
for item in my_list:
    print(item)
```

Iterating a specific number of times using range():

```
for i in range(5):  # Iterates from 0 to 4
    print(i)
```

Iterating with index and value using enumerate():

```
my_list = ['a', 'b', 'c']
for index, value in enumerate(my_list):
    print(f"Index: {index}, Value: {value}")
```

While Loops:
Repeating until a condition is false.

```
count = 0
while count < 3:
    print(count)
    count += 1
```

Loop Control Statements:

- break: Exits the loop entirely.

```
for i in range(10):
    if i == 5:
        break
    print(i)
```

- continue: Skips the rest of the current iteration and moves to the next.

```
for i in range(5):
    if i == 2:
        continue
    print(i)
```

Nested Loops:
Iterating through nested structures.

```
matrix = [[1, 2], [3, 4]]
for row in matrix:
    for element in row:
        print(element)
```

List Comprehensions (Concise Loop Alternative):

- Creating new lists based on existing ones:

```
numbers = [1, 2, 3, 4]
squared_numbers = [x**2 for x in numbers if x % 2 == 0]
# Result: [4, 16]
```

Class activity 4

Class Activity: Iterating through code with for and while

Objective:

To demonstrate understanding of loops in python.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.
For this activity you will need to create a list.
This can be done using the following code:

my_list = [ ] #To create a new list

1. Task 1
   - Use the appropriate loop, prompt the user to enter 10 numbers between 1 and 100.
   - Add those numbers to a list, and example of the code to add the numbers to the list:
   - my_list.append(int(input("Please enter a number between 1 and a 100"))
   - Print out the list to the user

2. Task 2
   - Comment out the previous print statement
   - Loop through the list and add all the numbers together
   - Print the sum of all the numbers

3. Task 3
   - Create a menu with the following options:
     - Enter numbers
     - Print numbers separately
     - Print the sum of the numbers
     - Clear the list
     - Close Application

4. Task 4
   - Using the appropriate loop change the application to be able to accept continuous input until the user closes the application

5. Task 5
   - Use an if statement of switch case to handle user input for the menu options

6. Task 6
   - Move the code from the previous tasks to the corresponding menu option

Homework: Facilitator to give additional homework activity

Revision Questions Day 4

1. In Python, what is the purpose of the range() function?
    A. It generates a sequence of numbers within a specified range without first having to create and store a large data structure.
    B. It checks whether a number is within a specified range.
    C. It generates a sequence of numbers within a specified range.
    D. It generates a random number within a specified range.

2. In Python, what is the function of the break statement within a loop?
    A. It checks whether the loop completed normally.
    B. It ends the loop prematurely.
    C. It skips to the next iteration of the loop.
    D. It can be used to end the loop prematurely if a certain condition is met.

3. What does the else block in a for loop execute?
    A. It executes before the loop starts.
    B. It never executes.
    C. It executes if the loop finishes normally, without a break.
    D. It executes after each iteration of the loop.

4. In Python, what happens when an else statement is used within a loop?
    A. The else statement is run if a break call was made.
    B. The else statement is ignored.
    C. The else statement is run if the loop completed normally without a break call.
    D. The else statement causes an error.

5. Consider a scenario where you need to print numbers from 1 to 10 but skip printing the number. Which Python loop implementation correctly achieves this? Select all that apply.
    A. for i in range(1, 11): if i == 5: continue else: print(i)
    B. while i < 11: if i == 5: continue i += 1 print(i)
    C. for i in range(1, 11): if i != 5: print(i)
    D. for i in range(1, 11): if i == 5: continue print(i)

6. What does for i in range(0, 11, 2) produce?
    A. Numbers from 0 to 10, skipping every second
    B. Even numbers from 0 to 10
    C. Odd numbers from 0 to 11
    D. All numbers from 0 to 11

7. When is the else block after a while or for loop executed?
    A. Always, after the loop
    B. Only if the loop runs at least once
    C. Only if the loop ends without a break
    D. Only if continue was used

8. What does the continue statement do inside a loop?
    A. Ends the loop entirely
    B. Skips the rest of the code in the current iteration and starts the next one
    C. Repeats the last iteration
    D. Skips the loop condition

9. Which statement correctly describes range(2, -1, -1)?
    A. It counts up from 2 to -1
    B. It returns [2, 1, 0]
    C. It produces an error
    D. It returns [2, 0, -1]

10. What does the following code print?

```
count = 1
while count <= 3:
    print(count)
    count += 1
```

    A. 111
    B. 123
    C. 1234
    D. Infinite loop

Week 2

Day 5 - Learning Unit 5

Introduction

In this chapter, we will look at lists, tuples, dictionaries and sets, as well as the different functions that can be applied to them. Students will also be introduced to functions in this section. The facilitator will cover slide 5 and is linked to textbook 1 chapter 7, 8, 9.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Programming basics

Additional Content

Accessing Items in Lists	
The table is based on this list:	**fruit_list = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"].**
Command	**Description**
fruit_list[0]	Get the first item on the list ("apple")
fruit_list[1]	Get the second item on the list ("banana")
fruit_list[-1]	Get the last item ("mango")
fruit_list[2:5]	Get the items from start to end indexes
fruit_list[:4]	Get the items from the beginning but exclude "kiwi" and beyond
fruit_list[2:]	Get the items from "cherry" to the end
fruit_list[4:-1]	Get the items from "orange" (-4) onwards but exclude "mango" (-1)
if "apple" in fruit_list: print("Yes, we have 'apple'")	Check if "apple" is in the list

Dictionary Operations		
COMMAND	**DESCRIPTION**	**EXAMPLE**
del dict1["key1"]	Remove the item with the specified key name	del meta["WA"] #"WhatsApp"
del dict1	Delete the dictionary	del meta
dict1[key1]	Access the value of a dictionary dict1 element using its key key1	meta["FB"] #"Facebook"
Dictionary method	**Description**	**Usage**
clear()	Remove all the elements from the dictionary	dict1.clear()
copy()	Return a copy of the dictionary	dict1.copy()
fromkeys()	Return a dictionary with the specified keys and value	dict1.fromkeys(keys, value)
get()	Return the value of the specified key	dictionary.get(key_name, value)
items()	Return a list containing a tuple for each key-value pair	dict1.items()
keys()	Return a list containing the dictionary's keys	dict1.keys()
pop()	Remove the element with the specified key	dict1.pop(key_name)
popitem()	Remove the last inserted key-value pair	dict1.popitem()
setdefault()	Return the value of the specified key. If the key does not exist, add as new key-value pair	dict1.setdefault(key_name, value)
update()	Update the dictionary with the specified key-value pairs	dict1.update(iterable)
values()	Return a list of all the values in the dictionary	dict1.values()

Class activity 5

Class Activity: Lists, Dictionaries and Functions

Objective:

To illustrate understanding of how to create and work with lists, dictionaries and functions.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.
Create a dictionary with the following values:

- harvest: the process or period of gathering in crops
- sew: join, fasten, or repair (something) by making stitches with a needle and thread or a sewing machine
- farm: an area of land and its buildings, used for growing crops and rearing animals
- arbitration: the use of an arbitrator to settle a dispute

1. Task 1
    - Create a function that will print out the following menu options
    - Enter dictionary item
    - Edit dictionary items
    - Print dictionary items
    - Remove dictionary items
    - Print dictionary keys
    - Print dictionary values
    - Clear dictionary
    - Close Application
    - The function should accept user input for the selected menu option and return the desired menu option

2. Task 2
    - Use the appropriate loop to allow the application to accept continuous input and print the main menu.
    - Use an if statement or switch case to navigate menu options

3. Task 3
    - For option 1, create a function that takes 2 arguments to add an item to the dictionary.
    - Prompt the user for a key and a value and pass those to values to the function.
    - The function should give feedback to the user that the new item was successfully added to the dictionary.

4. Task 4
    - In option 2, prompt the user to enter a key and the value to change.

5. Task 5
    - Create a function that prints out the dictionary key and value of all the dictionary items, each on its own line.
    - Call the function in option 3

6. Task 6
    - Create a function that accepts a dictionary key as an argument, that removes that key and value from the value.
    - In option 4, prompt the user for an item to delete from the dictionary and call the function, passing that key to the function

7. Task 7
    - Create a function that adds all the dictionary keys to a list and iterate through the list and print all the keys on a new line.
    - In option 5 call the function

8. Task 8
    - Create a function that adds all the dictionary values to a list and iterate through the list and print all the keys on a new line.
    - In option 6 call the function

9. Task 9
    - For option 7, clear the dictionary

10. Task 10
    - For option 0, close the application

11. Task 11
    - If the user types in any other number, display an error message

Homework: Facilitator to give additional homework activity

Revision Questions Day 5.

1. In Python, what happens when you try to modify a tuple?
    A. The tuple is duplicated.
    B. The tuple is automatically converted into a list.
    C. Python does not allow it because tuples are immutable.
    D. Python throws a syntax error.

2. In Python, what does the sort() method do to a list?
    A. Removes duplicate items from the list
    B. Returns a sorted copy of the list
    C. Reverses the order of the list
    D. Sorts the list itself, in place

3. In Python, what is the correct way to create a tuple with one or more elements?
    A. Enclose each element in curly brackets.
    B. Enclose each element in square brackets.
    C. Follow each element with a comma.
    D. Follow each element with a semicolon.

4. In Python, which data structure does not support indexing or keys?
    A. Tuple
    B. Set
    C. Dictionary
    D. List

5. What does the pop() method do in Python dictionaries?
    A. It removes a random item from the dictionary.
    B. It removes an item with the provided key but does not return its value.
    C. It removes all items from the dictionary.
    D. It removes an item with the provided key and returns its value.

6. In Python, which method is used to get all the keys in a dictionary?
    A. get()
    B. keys()
    C. items()
    D. values()

7. In Python, what does the code except Exception as other do?
    A. It ignores all exceptions.
    B. It raises an exception named other.
    C. It defines a new exception type named other.
    D. It catches all exceptions not caught by previous except blocks and assigns the exception object to the variable other.

8. In Python, what does it mean for functions to be first-class citizens?
    A. Functions have the same rights and privileges as human citizens.
    B. Functions can vote in Python elections.
    C. Functions can be assigned to variables, used as arguments to other functions, and returned from functions.
    D. Functions can own property in Python.

9. In Python, what does the double asterisk (**) do when used in a function?
    A. It creates a pointer to the function.
    B. It multiplies the function's output by two.
    C. It groups keyword arguments into a dictionary.
    D. It squares the function's output.

10. In Python, what happens if an error occurs inside a try block?
    A. The code inside the try block continues to run.
    B. The error is ignored.
    C. The program immediately terminates.
    D. The code inside the except block runs.

Day 6 - Learning Unit 6

Introduction:

In this chapter, we look at classes and object. Students will learn how to use classes and object in applications. The facilitator will cover slide 6 and is linked to textbook 1 chapter 10.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes:

- Software applications

Additional Resources

1. Classes:

- Definition: A class is a blueprint or a template for creating objects. It defines the attributes (data) and methods (functions) that objects of that class will possess.
- Syntax:

```
class ClassName:
    # Class attributes (shared by all instances)
    class_attribute = "Value"

    # Constructor method
    def __init__(self, param1, param2):
        self.param1 = param1  # Instance attribute
        self.param2 = param2  # Instance attribute

    # Instance method
    def instance_method(self):
        print(f"This is an instance method. Param1: {self.param1}")

    # Class method (operates on the class, not an instance)
    @classmethod
    def class_method(cls):
        print(f"This is a class method. Class attribute: {cls.class_attribute}")

    # Static method (does not operate on class or instance)
    @staticmethod
    def static_method():
        print("This is a static method.")
```

2. Objects (Instances):

- Definition: An object is an instance of a class, created from the class blueprint.
- Creation:

```
obj_name = ClassName(arg1, arg2)
```

Accessing Attributes and Methods.

```
print(obj_name.param1)
obj_name.instance_method()
```

3. Key Concepts:
self:

A reference to the current instance of the class. It is the first parameter in all instance methods and is used to access instance attributes and methods.

__init__ (Constructor):

A special method called automatically when a new object is created. It is used to initialize the object's attributes.

Inheritance:
Allows a class (child/derived class) to inherit attributes and methods from another class (parent/base class).

```
class ChildClass(ParentClass):
    pass
```

Polymorphism:
The ability of different objects to respond to the same method call in their own specific ways.

Encapsulation:
The bundling of data (attributes) and methods that operate on the data within a single unit (class), restricting direct access to some of the object's components (using private/protected variables).

- Protected variables: Start with a single underscore (e.g., _protected_var). Conventionally, they should not be accessed directly from outside the class or its subclasses.
- Private variables: Start with two underscores (e.g., __private_var). Python performs name mangling to make them harder to access directly from outside the class.

4. Special Methods (Dunder Methods):
Methods with double underscores (e.g., __str__, __len__, __add__) that define how objects behave in certain contexts (e.g., string representation, length, addition).

Example Usage:

```
class Dog:
    species = "Canis familiaris"  # Class attribute

    def __init__(self, name, breed):
        self.name = name  # Instance attribute
        self.breed = breed  # Instance attribute

    def bark(self):
        print(f"{self.name} says Woof!")

    @classmethod
    def get_species(cls):
        print(f"Species: {cls.species}")

my_dog = Dog("Buddy", "Golden Retriever")
print(my_dog.name)
my_dog.bark()
Dog.get_species()
```

Class activity 6

Class Activity: Working with Classes and Object

Objective:

To illustrate and understanding of how to work with classes and objects.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

1. Task 1
   - Create 2 python files named:
     - StudentClass
     - StudentMain

2. Task 2
   - In the StudentClass file:
   - Create a Base Class called Student with:
     - Attributes:
       - name (string)
       - age (integer)
       - grade (string, e.g., "A", "B", "C")
     - Methods:
       - display_info() → prints all details of the student
       - is_pass() → returns True if grade is "A", "B", or "C", else False
   - Create a Subclass called GraduateStudent that:
     - Inherits from Student.
     - Adds a new attribute: degree (string, e.g., "BSc Computer Science").
     - Overrides display_info() to also show the degree.

3. Task 3
   - In the StudentMain file
   - Create Objects
     - Create two Student objects and two GraduateStudent objects
     - Store them in a list

4. Task 4
   - Loop Through Students
     - Use a loop to display each student's info (use polymorphism so the correct display_info() is called).
     - Show whether each student passed

Homework: Facilitator to give additional homework activity

Revision Questions Day 6

1. In Python, what is a class method?
    A. A method that is part of the parent class
    B. A method that is part of the class itself
    C. A method that is part of the individual objects created from the class
    D. A method that is not part of the class or its objects

2. In the class Element, what are the instance attributes?
    A. 'name', 'symbol', 'number'
    B. 'Laser', 'Claw', 'SmartPhone'
    C. 'abc', 'xyz', '123'
    D. 'berries', 'clover', 'campers'

3. In object-oriented programming, what is the term for a class that is derived from more than one parent class?
    A. Child class
    B. Parent class
    C. Multiple inheritance
    D. Subclass

4. In Python, what is a static method?
    A. A method that is not part of the class or its objects
    B. A method that is part of the class itself
    C. A method that is part of the parent class
    D. A method that is part of the individual objects created from the class

5. In Python, what is the function of the __str__() special method?
    A. It is used to initialize the object.
    B. It is used to calculate the length of the object.
    C. It is used to represent the object as a string.
    D. It is used to print the object.

6. In Python, what is the concept of applying the same operation to different objects, based on the method's name and arguments, regardless of their class?
    A. Class methods
    B. Magic methods
    C. Static methods
    D. Duck typing

7. In Python, what is the difference between aggregation and composition?
    A. In composition, one thing is part of another, while in aggregation, one thing uses another.
    B. Aggregation involves inheritance, while composition does not.
    C. Composition involves inheritance, while aggregation does not.
    D. In aggregation, one thing is part of another, while in composition, one thing uses another.

8. In Python, what is the purpose of the self argument in instance methods?
    A. It refers to the individual object itself.
    B. It has no specific purpose and is just a convention.
    C. It refers to the parent class.
    D. It refers to the class itself.

9. In object-oriented programming, what function is used to call a method from a parent class in a child class?
    A. super()
    B. issubclass()
    C. subclass()
    D. init()

10. In Python, what does the @classmethod decorator indicate?
    A. The following function is a class method.
    B. The following function is a magic method.
    C. The following function is a static method.
    D. The following function is an instance method.

Day 7 - Learning Unit 7

Introduction

In this chapter, we look at modules and packages in Python. Students will also be introduced to Unicode, regular expressions and binary data in Python. Students will learn how to work with date and time in Python The facilitator will cover slide 7 and is linked to textbook 1 chapters 11, 12 and 13.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Software applications

Additional Resources

1. Importing:

```
from datetime import date, time, datetime, timedelta
```

2. Creating Date and Time Objects:

- date.

```
d = date(2025, 8, 5) # Year, Month, Day
```

- time.

```
t = time(12, 45, 30) # Hour, Minute, Second
```

- datetime.

```
dt = datetime(2025, 8, 5, 12, 45, 30) # Year, Month, Day, Hour, Minute, Second
```

- Current Date/Time.

```
current_date = date.today()
current_datetime = datetime.now()
```

3. Accessing Components:

```
year = dt.year
month = dt.month
day = dt.day
hour = dt.hour
minute = dt.minute
second = dt.second
```

4. Formatting (datetime to string): strftime()

Directive	Description	Example (for datetime(2025, 8, 5, 12, 45, 30))
%Y	Year with century	2025
%m	Month as zero-padded decimal	08
%d	Day as zero-padded decimal	05
%H	Hour (24-hour clock)	12
%M	Minute as zero-padded decimal	45
%S	Second as zero-padded decimal	30
%A	Weekday full name	Monday
%B	Month full name	August
%c	Locale's appropriate date and time representation	Mon Aug 5 12:45:30 2025

Example:

```
formatted_dt = dt.strftime("%Y-%m-%d %H:%M:%S") # "2025-08-05 12:45:30"
```

5. Parsing (string to datetime): strptime()

```
date_string = "2025-08-05 12:45:30"
parsed_dt = datetime.strptime(date_string, "%Y-%m-%d %H:%M:%S")
```

6. Time Differences: timedelta

```
diff = timedelta(days=7, hours=3)
new_dt = dt + diff # Add a timedelta
old_dt = dt - diff # Subtract a timedelta
```

Class activity 7

Class Activity: Working with Time and Date in Python

Objective:

To demonstrate an understanding of how to work with date and time functions in Python using the proper libraries.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

- Create a Python file named time_date_activity.py.
- Complete each task below.

1. Task 1
   - Current Date and Time
     - Write a program that prints the current date and current time.
     - Format the output as:

       Today is: Monday, 11 August 2025
       Current time: 13:45:20

2. Task 2
   - Birthday Countdown
     - Ask the user to input their birthday in the format YYYY-MM-DD.
     - Calculate how many days are left until their next birthday.
     - Print:

       Your next birthday is in X days!

3. Task 3
   - Date Arithmetic
     - Display today's date.
     - Calculate and print the date 100 days from today.

4. Task 4
   - String to Date Conversion
     - Given the string "2025-12-25", convert it to a datetime object and print:

       Christmas 2025 is on a Thursday

5. Task 5
    - Stopwatch Simulation
        - Record the start time.
        - Wait for 3 seconds using time.sleep(3).
        - Record the end time.
        - Print the elapsed time in seconds.

        Today is: Monday, 11 August 2025
        Current time: 14:03:45
        Enter your birthday (YYYY-MM-DD): 2025-12-01
        Your next birthday is in 112 days!
        Today's date: 2025-08-11
        The date 100 days from today: 2025-11-19
        Christmas 2025 is on a Thursday
        Elapsed time: 3.002 seconds

Revision Questions Day 7

1. In Python, how can you import a module with another name?
    A. Use the change statement
    B. Use the module...to..``. statement
    C. Use the import...as..``. statement
    D. Use the rename statement

2. What does the Python function pprint() do?
    A. It prints a sequence in a random order.
    B. It prints a sequence in reverse order.
    C. It prints a sequence in the order it was added.
    D. It prints elements in a way that aligns them for better readability.

3. What is the purpose of the __init__.py file in a Python package?
    A. It contains the main function of the package.
    B. It indicates that the directory should be treated as a package.
    C. It contains the configuration settings for the package.
    D. It initializes the Python interpreter.

4. In Python, which method is used to handle encoding exceptions by substituting '?' for unknown characters?
    A. strict
    B. ignore
    C. replace
    D. backslashreplace

5. In Python, what is the function used to encode a string to bytes?
    A. transform()
    B. decode()
    C. encode()
    D. convert()

6. In Python, what is the standard text encoding?
    A. ASCII
    B. UTF-8
    C. Windows 1252
    D. Latin-1

7. In Python's datetime module, which object class is used to represent a time of day?
    A. datetime
    B. date
    C. timedelta
    D. time

8. Which function in Python's time module can be used to convert epochs to strings?
    A. strftime()
    B. strptime()
    C. ctime()
    D. isoformat()

9. What does the now() method in Python's datetime module return?
    A. The current date and time
    B. The current time zone
    C. The current time
    D. The current date

10. What happens if the date string you want to parse with strptime() has spaces instead of dashes?
    A. An exception is raised.
    B. The function will ignore the spaces.
    C. The function will still parse the date string correctly.
    D. The function will automatically replace the spaces with dashes.

Day 8 - Learning Unit 8

Introduction

In this chapter, we look at using files and directories in Python. Students will learn how to access data stored in files and write data to files. Learners will also learn how to navigate folders using Python. The facilitator will cover slide 8 and is linked to textbook 1 chapter 14.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at:
https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes:

- Software applications

Additional Resources

Common operations using the os and shutil modules, as well as built-in file handling functions.
File Operations:

Opening a file.

```
f = open("filename.txt", "mode") # 'r' (read), 'w' (write), 'a' (append), 'x' (create), 'b' (binary), '+' (read/write)
```

Reading from a file.

```
content = f.read() # Read entire file
line = f.readline() # Read one line
lines = f.readlines() # Read all lines into a list
```

Writing to a file.

```
f.write("text to write")
```

Closing a file.

```
f.close()
```

Using with statement (recommended for automatic closing):

```
with open("filename.txt", "r") as f:
    content = f.read()
```

Directory Operations (using os module):
Importing the os module.

```
import os
```

Getting current working directory.

```
current_dir = os.getcwd()
```

Changing directory.

```
os.chdir("new_directory_path")
```

Listing directory contents.

```
contents = os.listdir("directory_path")
```

Creating a directory.

```
os.mkdir("new_directory_name")
```

Creating directories recursively.

```
os.makedirs("path/to/new/directories")
```

Removing a file.

```
os.remove("filename.txt")
```

Removing an empty directory.

```
os.rmdir("empty_directory_name")
```

Checking if path exists.

```
exists = os.path.exists("path/to/file_or_directory")
```

Checking if it's a file.

```
is_file = os.path.isfile("path/to/file")
```

Checking if it's a directory.

```
is_dir = os.path.isdir("path/to/directory")
```

Joining paths.

```
full_path = os.path.join("dir1", "dir2", "filename.txt")
```

Advanced File/Directory Operations (using shutil module):
Importing the shutil module.

```
import shutil
```

Copying a file.

```
shutil.copy("source_file.txt", "destination_file.txt")
```

Copying a directory recursively.

```
shutil.copytree("source_directory", "destination_directory")
```

Moving/Renaming a file or directory.

```
shutil.move("source_path", "destination_path")
```

Removing a directory and its contents.

```
shutil.rmtree("directory_to_remove")
```

Class activity 8

Class Activity: Working with Files and Directories in Python

Objective:

To demonstrate an understanding of working with files and directories in Python.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

- Create a Python file named files_dirs_activity.py.
- Complete each task below.

1. Task 1
   - Create a new directory called student_data in the current working directory.
   - Print the path to confirm it was created.

2. Task 2
   - Inside student_data, create a file called students.txt.
   - Write the following names into the file (one per line):

     Alice
     Bob
     Charlie
     Diana

3. Task 3
   - Open students.txt and print each name with the text:

     Student: Alice

4. Task 4
   - Add two more names to students.txt:

     Ethan
     Fiona

5. Task 5
   - Rename students.txt to class_list.txt.

6. Task 6
   - Create a new folder called backup.
   - Move class_list.txt into the backup folder.

7. Task 7
    - Delete class_list.txt from backup.
    - Delete the empty backup folder.

    #Sample Output
    Created directory: /path/to/student_data
    Created file: students.txt
    Student: Alice
    Student: Bob
    Student: Charlie
    Student: Diana
    Appended new students.
    Renamed file to class_list.txt
    Moved file to backup folder.
    Deleted file and backup folder.

Homework: Facilitator to give additional homework activity

Revision Questions Day 8

1. In Python, what does the 'w' mode indicate when opening a file?
    A. Append
    B. Binary
    C. Write
    D. Read

2. What is the purpose of the os.path.join() function in Python?
    A. To join two or more files into a single file
    B. To combine two or more pathnames with the proper path separation character for your operating system
    C. To combine two or more pathnames into a single path
    D. To join two or more directories into a single directory

3. In Python, what does the 'b' mode indicate when opening a file?
    A. Binary
    B. Write
    C. Read
    D. Append

4. What does the os.path.exists() function do in Python?
    A. It checks if a specific path is absolute.
    B. It checks if a specific path is a file.
    C. It checks if a specific path exists.
    D. It checks if a specific path is a directory.

5. In Python, what is the correct way to open a file for reading in binary mode?
    A. open('bfile', 'rb')
    B. open('bfile', 'w')
    C. open('bfile', 'r')
    D. open('bfile', 'wb')

6. In Python, what does the 'r' mode indicate when opening a file?
    A. Read
    B. Binary
    C. Write
    D. Append

7. What is the purpose of the seek() function in Python?
    A. To find a specific line in a file
    B. To replace a specific byte in a file
    C. To jump to another byte offset in the file
    D. To delete a specific byte in a file

8. In Python, what does the 'a' mode indicate when opening a file?
    A. Read
    B. Binary
    C. Write
    D. Append

9. In Python, what does the 'a+' mode indicate when opening a file?
    A. Read and write
    B. Binary read and write
    C. Write and read
    D. Append and read

10. In Python, what does the 'a+' mode indicate when opening a file?
    A. Read and write
    B. Binary read and write
    C. Write and read
    D. Append and read

Week 3

Day 9 - Learning Unit 9

Introduction

In this chapter, we look at Concurrency, queues, processes and threads. Students will learn about creating applications with multiple threads. The facilitator will cover slide 9 and is linked to textbook 1 chapter 15.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes:

- Software applications

Additional Resources

Common functionalities of the threading module for concurrent programming.
Core Concepts:

- threading.Thread: The fundamental class for creating and managing threads.
    - Creating a thread:

```
import threading

def my_function():
    # Code to be executed in the thread
    pass

t = threading.Thread(target=my_function, args=(arg1, arg2), kwargs={'key': 'value'})
```

- Subclassing threading.Thread.

```
import threading

class MyThread(threading.Thread):
def run(self):
        # Code to be executed when the thread starts
        pass

my_thread_instance = MyThread()
```

- Starting a thread: t.start() – Invokes the run() method (or target function) in a separate thread of control.
- Joining a thread: t.join([timeout]) – Blocks the calling thread until the target thread terminates. timeout specifies a maximum wait time in seconds.
- Daemon threads: t.daemon = True – A daemon thread will not prevent the program from exiting if all non-daemon threads have finished.
- is_alive(): t.is_alive() – Returns True if the thread is currently executing.
- current_thread(): threading.current_thread() – Returns the currently active Thread object.

Synchronization Primitives:

- threading.Lock: A basic mutual exclusion lock to protect critical sections and prevent race conditions.

```
lock = threading.Lock()
with lock:
        # Critical section
        pass
```

- threading.RLock: A reentrant lock that can be acquired multiple times by the same thread.

- threading.Semaphore: Limits the number of threads that can access a resource concurrently.

```
semaphore = threading.Semaphore(value=3) # Allow 3 concurrent accesses
with semaphore:
    # Access limited resource
    pass
```

- threading.Event: A simple flag that threads can wait for or set.

```
event = threading.Event()
# In one thread:
event.wait() # Blocks until event is set
# In another thread:
event.set() # Unblocks waiting threads
```

- threading.Condition: Allows threads to wait for specific conditions to be met.
- threading.threading.Barrier: Synchronizes multiple threads at a common point, ensuring all threads reach the barrier before proceeding.

Common Use Cases:

- I/O-bound tasks:
  - Ideal for tasks like network requests, file operations, or database interactions, where threads can wait for external resources without blocking the main program.

- Background processing:
  - Running tasks in the background without affecting the responsiveness of the main application.

Class activity 9

Class Activity: Introduction to Threading in Python

Objective:

To demonstrate an understanding of threads in Python.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

- Create a Python file named threading_activity.py.
- Complete each task below.

1. Task 1
   - Import the threading and time modules.
   - Print the main thread's name.

2. Task 2
   - Create a function print_numbers() that:
     - Prints numbers from 1 to 5.
     - Waits 1 second between prints.
3. Task 3
   - Create a function print_letters() that:
     - Prints letters from A to E.
     - Waits 1 second between prints.

4. Task 4
   - Call print_numbers() and print_letters() normally.
   - Note the total time taken.

5. Task 5
   - Create two threads:

     ```
     t1 = threading.Thread(target=print_numbers)
     t2 = threading.Thread(target=print_letters)
     ```

   - Start both threads.
   - Use .join() to wait for them to finish.
   - Note the total time taken.

6. Task 6
   - Print the time taken in both cases and explain why threading is faster.

Homework: Facilitator to give additional homework activity

Revision Questions Day 9

1. What is the recommended use of threads and processes in Python?
    A. Avoid using threads and processes in Python
    B. Use threads for I/O-bound problems and processes for CPU-bound problems
    C. Use threads and processes interchangeably for any problem
    D. Use threads for CPU-bound problems and processes for I/O-bound problems

2. In the context of concurrency in Python, what is the difference between synchronous and asynchronous tasks?
    A. There is no difference between synchronous and asynchronous tasks.
    B. Synchronous tasks follow one after the other, while asynchronous tasks are independent.
    C. Synchronous tasks are slower than asynchronous tasks.
    D. Synchronous tasks are independent, while asynchronous tasks follow one after the other.

3. In Python, which module can be used to retrieve system and process information?
    A. subprocess
    B. multiprocessing
    C. terminate
    D. os

4. In Python, what is the Global Interpreter Lock (GIL)?
    A. A function that allows for faster processing of data
    B. A tool that enhances the performance of Python threads
    C. An implementation detail that can make a multithreaded program slower than its single-threaded counterpart
    D. A module that speeds up CPU-bound tasks

5. What does the getoutput() function in Python's subprocess module do?
    A. It retrieves information about a process.
    B. It stops a running process.
    C. It runs another program and captures its output.
    D. It starts a new process.

6. Which Python library is used to create concurrent processes?
    A. multiprocessing
    B. gevent
    C. Redis
    D. asyncio

7. What is the role of the Redis library?
    A. It is used to create concurrent processes.
    B. It is used to create a user interface.
    C. It is used to manage data flow between processes.

D. It is used to handle network connections.

8. What does the concurrent.futures module in Python do?
   A. It allows for the creation of multiple threads or processes without any additional details.
   B. It speeds up the execution of Python programs.
   C. It allows for the creation of a single-threaded program.
   D. It simplifies the use of threads or multiple processes by scheduling an asynchronous pool of workers.

9. What is the purpose of the terminate() function in Python's multiprocessing module?
   A. To pause a running process
   B. To retrieve information about a process
   C. To create a new process
   D. To stop a running process

10. True or False: Python is the only language to use threading.
    A. True
    B. False

Day 10 - Learning Unit 10

Introduction

In this chapter, we look at persistent data storage in programming. Students will be introduced to SQL, and also some of the basic CRUD operations. The facilitator will cover slide 10 and is linked to textbook 1 chapter 16 and textbook 2 chapters 1, 2, and 3.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Software applications

Additional Resources

1. Create (Insert Data)
Purpose: Adds new rows (records) into a table.
Syntax:

```
INSERT INTO table_name (column1, column2, ...)
VALUES (value1, value2, ...);
```

Example:

```
INSERT INTO Customers (CustomerID, FirstName, LastName)
VALUES (1, 'Alice', 'Smith');
```

2. Read (Retrieve Data)
Purpose: Retrieves data from one or more tables.
Syntax:

```
SELECT column1, column2, ...
FROM table_name
WHERE condition; -- Optional: filters rows
ORDER BY column_name ASC|DESC; -- Optional: sorts results
```

Example:

```
SELECT FirstName, LastName
FROM Customers
WHERE CustomerID = 1;
```

3. Update (Modify Data)
Purpose: Modifies existing data within a table.
Syntax:

```
UPDATE table_name
SET column1 = new_value1, column2 = new_value2, ...
WHERE condition; -- Crucial: specifies which rows to update
```

Example:

```
UPDATE Customers
SET LastName = 'Jones'
WHERE CustomerID = 1;
```

4. Delete (Remove Data)
Purpose: Removes rows (records) from a table.
Syntax:

```
DELETE FROM table_name
WHERE condition; -- Crucial: specifies which rows to delete
```

Example:

```
DELETE FROM Customers
WHERE CustomerID = 1;
```

Class activity 10

Class Basic SQL Queries

Objective:

To illustrate an understanding of basic SQL.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

1. Task 1
   - Create a Table using the following information:
     - id Primary Key Auto Increment
     - first_name
     - last_name
     - age
     - grade

2. Task 2
   - Insert the following Sample Data:
     - Alice, Smith, 14, 9th
     - Bob, Johnson, 15, 10th
     - Charlie, Lee, 14, 9th
     - Diana, Brown, 16, 11th
     - Ethan, Davis, 15, 10th

3. Task 3
   - Write an SQL query to select all columns and rows from the students table.

4. Task 4
   - Write an SQL query to select only first_name and grade from all students.

5. Task 5
   - Write an SQL query to find all students who are in the 10th grade.

6. Task 6
   - Write an SQL query to select all students ordered by last_name ascending.

7. Task 7
   - Write an SQL query to update Ethan's grade from 10th to 11th.

8. Task 8
    - Write an SQL query to delete students who are 16 years old.

9. Task 9
    - Write an SQL query to count how many students are in 9th grade.

Homework: Facilitator to give additional homework activity

Revision Questions Day 10

1. In Python programming, which module is used to handle Windows-style .ini files?
    A. csv
    B. configparser
    C. mmap
    D. h5py

2. In Python programming, what is the purpose of the SQL language?
    A. To handle .ini files
    B. To manage data in relational databases
    C. To handle CSV files
    D. To manage binary files

3. In Python's SQLAlchemy library, what does the execute() function do?
    A. It deletes a database.
    B. It runs one or more SQL commands against the database.
    C. It closes the connection to a database.
    D. It creates a new database.

4. In Python programming, what is the purpose of the DB-API?
    A. To access relational databases
    B. To handle CSV files
    C. To handle Windows-style *.ini *files
    D. To manage binary files

5. What is the purpose of the xml.etree.ElementTree module in Python?
    A. To parse XML files
    B. To delete XML files
    C. To rename XML files
    D. To create XML files

6. What is the purpose of the csv.writer() function in Python's csv module?
    A. To open a CSV file
    B. To read data from a CSV file
    C. To write data to a CSV file
    D. To close a CSV file

7. In the context of Python libraries, what is the function of the json.dumps() method?
    A. It converts a JSON string into a Python data structure.
    B. It checks if a Python data structure can be converted into a JSON string.
    C. It converts a Python data structure into a JSON string.
    D. It checks if a JSON string is valid.

8. What is the purpose of the fetchall() function in Python's SQLAlchemy library?
    A. To insert new records into a database
    B. To retrieve all results from a previously executed SQL command
    C. To delete all records from a database
    D. To update all records in a database

9. In the context of Python's SQLAlchemy library, what is a placeholder?
    A. A variable that stores the result of a SQL query.
    B. A symbol in a SQL command that will be replaced with actual data when the command is executed.
    C. A function that pauses the execution of a SQL command.
    D. A temporary database for testing purposes.

10. In the context of data storage, what is a record?
    A. A method for storing data
    B. A chunk of related data
    C. A type of file format
    D. A type of data structure

Day 11 - Learning Unit 11

Introduction

In this chapter, students will be introduced to using Python in a network and web environment. The facilitator will cover slide 11 and is linked to textbook 1 chapter 17 and 18.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Software applications

Additional Resources

1. Sockets (the socket module):

- Creating Sockets.

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) # TCP socket
udp_s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # UDP socket
```

- Server-side.

```
s.bind(('localhost', 12345)) # Bind to address and port
s.listen(5) # Listen for connections
conn, addr = s.accept() # Accept a connection
```

- Client-side.

```
s.connect(('example.com', 80)) # Connect to a remote host and port
```

- Sending/Receiving Data.

```
conn.sendall(b'Hello, server!') # Send data
data = conn.recv(1024) # Receive data
```

- Closing Sockets.

```
s.close()
```

2. Higher-Level Protocols (e.g., requests for HTTP):

- Making HTTP GET Requests.

```
import requests
response = requests.get('http://www.example.com')
print(response.text)
```

- Making HTTP POST Requests.

```
data = {'key': 'value'}
response = requests.post('http://www.example.com/api', json=data)
```

3. Asynchronous Programming (asyncio):

- Basic Async/Await.

```
import asyncio

async def my_coroutine():
    print("Starting...")
    await asyncio.sleep(1)
    print("Finished.")

asyncio.run(my_coroutine())
```

4. Network Information:

- Getting Hostname and IP.

```
import socket
hostname = socket.gethostname()
ip_address = socket.gethostbyname(hostname)
```

- DNS Resolution.

```
import socket
ip_addresses = socket.gethostbyname_ex('www.google.com')[2]
```

5. Common Libraries:

- socket: Low-level network communication.
- requests: Simplified HTTP client.
- http.server: Simple HTTP server for local development.
- asyncio: Asynchronous I/O framework.
- paramiko: SSH client and server.

Class Activity 11

Class Activity: Basic Web and Network Programming in Python

Objective:

To demonstrate an understanding of web and networking in Python.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

- requests library installed (pip install requests) required

1. Task 1: Create a TCP Server
   - Create a Python script named tcp_server.py.
   - Use the socket module to:
     - Bind to localhost on port 65432
     - Listen for a connection
     - Accept a connection and receive data
     - Print received data
     - Send a response message like "Hello from server!"
     - Close the connection

2. Task 2: Create a TCP Client
   - Create a Python script named tcp_client.py.
   - Use the socket module to:
     - Connect to the server at localhost on port 65432
     - Send a message, e.g. "Hello from client!"
     - Receive and print the server's response
     - Close the connection

3. Task 3: HTTP Request with requests
   - Create a script named http_request.py.
   - Use the requests library to:
     - Make a GET request to https://api.github.com
     - Print the HTTP status code
     - Print the value of the Content-Type header
     - Print the first 300 characters of the response content

Homework: Facilitator to give additional homework activity

Revision Questions Day 11

1. In Python networking, what type is used because messages are bytes, not Unicode text strings?
    A. Integer
    B. Float
    C. String
    D. Bytes

2. What is a Remote Procedure Call (RPC) in Python?
    A. It is a function that executes on remote machines across a network and looks like a normal function.
    B. It is a function that executes on remote machines across a network and registers and gets a key for accessing data.
    C. It is a function that executes on remote machines across a network.
    D. It is a function that executes on remote machines across a network and limits request traffic to servers.

3. If you were to compare the functionalities of Salt and Ansible to another tool, which tool would that be?
    A. Fabric
    B. Twirp
    C. Hadoop
    D. gRPC

4. What does the pickle module in Python do?
    A. It allows for data serialization in a special binary format.
    B. It allows for data serialization in a special binary format and can register and get a key for accessing data.
    C. It allows for data serialization in a special binary format and can save and restore any object.
    D. It allows for data serialization in a special binary format and can limit request traffic to servers.

5. In the context of networking, what does User Datagram Protocol (UDP) not guarantee?
    A. The delivery of the messages
    B. The speed of the messages
    C. The security of the messages
    D. The size of the messages

6. In the context of Python web scraping and automation, what is the function of the BeautifulSoup module?
    A. It is used to provide data through a web API.
    B. It is used to create automatic web pages for the typical database CRUD functions.
    C. It is used to extract data from HTML data from a website.
    D. It is used to automate a web fetcher, also known as a crawler or spider.

7. What is the purpose of the webbrowser module in Python web scraping and automation?
    A. It is used to direct your browser to a specific link.
    B. It is used to create a new resource on a web page.
    C. It is used to replace an existing resource on a web page.
    D. It is used to partially update a resource on a web page.

8. When building a basic website using Flask, what is the default port for the server to start up on?
    A. 3000
    B. 4000
    C. 6000
    D. 5000

9. In the process of building a basic website using Flask, what is the name of the Jinja2 template file created?
    A. index.html
    B. main.html
    C. base.html
    D. home.html

10. What is the purpose of the read() method in Python when extracting data from a webpage?
    A. To get the actual data contents from the webpage
    B. To delete data from the webpage
    C. To write data to the webpage
    D. To update data on the webpage

Day 12 - Learning Unit 12

Introduction

This chapter covers creating and working with a GUI using Python. The chapter covers topics such as using 2D and 3D graphics and Graphical User interfaces. The facilitator will cover slide 12 and is linked to textbook 1 chapter 20.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcomes

- Software applications

Additional Resources

Tkinter development.

Core Concepts:

- Importing: import tkinter or from tkinter import * (for basic widgets) and from tkinter.ttk import * (for themed widgets).
- Main Window: root = tk.Tk() creates the main application window.
- Event Loop: root.mainloop() starts the event handling loop, keeping the window open and responsive.

Common Widgets:

Informative:

- tk.Label: Displays static text or images.
- tk.Message: Displays static multi-line text.

Interactive:

- tk.Button: Triggers an action when clicked.
- tk.Entry: Allows single-line text input.
- tk.Text: Allows multi-line text input and display.
- tk.Checkbutton: A checkbox for boolean input.
- tk.Radiobutton: Allows selection of one option from a group.
- tk.Scale: A slider for selecting a numerical value.

Grouping:

- tk.Frame: A rectangular area for organizing other widgets.
- tk.LabelFrame: A frame with a descriptive label.
- tk.PanedWindow: A container that allows resizing its panes.
- Advanced (ttk):
- ttk.Treeview: Displays hierarchical data in a tree-like structure.
- ttk.Progressbar: Shows the progress of an operation.

Geometry Managers (Positioning Widgets):

- .pack(): Arranges widgets in blocks before placing them in the parent widget.
- .grid(): Arranges widgets in a two-dimensional grid using rows and columns.
- .place(): Positions widgets at specific coordinates within the parent widget.

Event Handling:

- command attribute:
  - Used with widgets like Button to specify a function to be called when the widget is interacted with.
- .bind() method:
  - Allows associating events (e.g., mouse clicks, key presses) with specific functions.

Example Structure:

```
import tkinter as tk
from tkinter import ttk

def button_click():
        print("Button clicked!")

# Create the main window
root = tk.Tk()
root.title("My Tkinter App")

# Create a label
label = tk.Label(root, text="Hello, Tkinter!")
label.pack(pady=10)

# Create a button
button = ttk.Button(root, text="Click Me", command=button_click)
button.pack(pady=5)

# Start the event loop
root.mainloop()
```

Class activity 12

Class Activity: Simple Tkinter Login Form

Objective:

To demonstrate an understanding of GUI in Python using Tkinter.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

1. Task 1
   - Create a Tkinter window titled "Login Form".
2. Task 2
   - Add two labels: "Username" and "Password".
3. Task 3
   - Add two text entry boxes:
     - One for username.
     - One for password (masked with show="*").
4. Task 4
   - Add a "Login" button.
5. Task 5
   - When the button is clicked:
     - If the username is "admin" and password is "1234", display "Login Successful!".
     - Otherwise, display "Invalid credentials!".

Homework: Facilitator to give additional homework activity

Revision Questions Day 12

1. Which Python library is used for creating graphical user interfaces and is part of the standard library?
    A. GTK+
    B. tkinter
    C. WxPython
    D. Qt

2. Which Python library is used for 2-D image processing and is not part of the standard library?
    A. Pillow
    B. turtle
    C. colorsys
    D. imghdr

3. True or false: Bokeh is a Python library that emphasizes quick visualization of large data sets and runs mainly in the browser.
    A. True
    B. False

4. In the context of web applications, which statement is true?
    A. A thick client and a thin client perform the same amount of work.
    B. A thick client lets the backend do most of the work.
    C. A thin client lets the backend do most of the work.
    D. A thin client performs all the work, while a thick client performs none.

5. What is the primary purpose of the imghdr module in Python's standard library?
    A. To edit and crop images
    B. To convert images between formats
    C. To detect the type of an image file
    D. To display images in a new window

6. What is the purpose of the putalpha() method in Pillow?
    A. Converts RGB to grayscale
    B. Adds transparency to an image
    C. Inverts the colors
    D. Resizes the image

7. Which library must be installed to use Pillow's Image.show() method effectively on most systems?
    A. Matplotlib
    B. Wand
    C. ImageMagick
    D. OpenCV

8. What is the purpose of the range() function in Python graphics or plotting?
    A. To calculate pixel depth
    B. To generate a sequence of numbers
    C. To crop images
    D. To loop through colors

9. Which of the following is a 3-D animation tool that comes with its own version of Python?
    A. Seaborn
    B. Bokeh
    C. Blender
    D. Matplotlib

10. Which Python package is best known for 2-D data visualization and plotting?
    A. Pillow
    B. Matplotlib
    C. Tkinter
    D. PySimpleGUI

Day 13 - Learning Unit 13

Introduction:

In this chapter, we look at strategies used for AI and Machine learning in Python. This chapter will introduce students to some of the basics of AI and Machine Learning. The facilitator will cover slide 13 and is linked to textbook 1 chapter 22.

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/

Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/

Learning outcome:

- Software applications

Additional Resources

Common tasks, concepts, and code snippets used in machine learning with Python. Key areas typically covered include:

1. Data Handling and Preprocessing:

- Libraries: pandas for data manipulation (DataFrames), numpy for numerical operations.
- Tasks: Loading data, handling missing values, encoding categorical features, scaling numerical features (StandardScaler, MinMaxScaler), splitting data (train_test_split).

2. Model Selection and Training:

- Libraries: scikit-learn for a wide range of algorithms.
- Supervised Learning:
    - Regression: Linear Regression, Ridge, Lasso, Decision Tree Regressor, Random Forest Regressor.
    - Classification: Logistic Regression, Support Vector Machines (SVC), Decision Tree Classifier, Random Forest Classifier, K-Nearest Neighbors (KNeighborsClassifier).
- Unsupervised Learning: K-Means Clustering, Principal Component Analysis (PCA).
- Training: model.fit(X_train, y_train).

3. Model Evaluation:

- Libraries: scikit-learn.metrics.
- Regression Metrics: Mean Squared Error (MSE), R-squared.
- Classification Metrics: Accuracy, Precision, Recall, F1-score, Confusion Matrix, ROC AUC.

4. Deep Learning (Briefly):

- Libraries: TensorFlow, Keras, PyTorch.
- Concepts: Neural Networks, layers (Dense, Conv2D, MaxPooling2D), activation functions, optimizers, loss functions.

5. Visualization:

- Libraries: matplotlib, seaborn.
- Tasks: Plotting distributions, scatter plots, heatmaps, bar charts.

Example Code Snippet (Scikit-learn Classification):

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

import pandas as pd

# Assuming 'df' is your DataFrame and 'target' is your target column
X = df.drop('target', axis=1)
y = df['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

Class activity 13

Class Activity: Math & Fractions Calculator

Objective:

To demonstrate an understanding of using the math and fractions functions in Python.

Instructions:

Students will have to use Python and VSCode to perform the following tasks.

1. Task 1
   - Import math and fractions modules.

2. Task 2
   - Ask the user to:
     - Enter two integers (for math operations).
     - Enter a numerator and denominator for two fractions (for fraction operations).

3. Task 3
   - Math module:
     - Square root of the first integer.
     - Factorial of the second integer.
     - Power calculation (first number ^ second number).
     - Greatest common divisor (GCD).

   - Fractions module:
     - Sum, difference, product, and quotient of the two fractions.

4. Task 4
   - Display results clearly.

Homework: Facilitator to give additional homework activity

Revision Questions Day 13

1. In NumPy, how can you create an array where all the values are one?
    A. Using the random() method.
    B. Using the zeros() method.
    C. Using the arange() method.
    D. Using the ones() method.

2. You are working with complex numbers in Python. Which module should you use?
    A. fractions
    B. math
    C. decimal
    D. cmath

3. Which of the following tasks can be performed using the Pandas library in Python? Select all that apply.
    A. Joining data
    B. Splitting data
    C. Reshaping data

4. What is the base data structure defined by the Pandas library in Python?
    A. DataFrame
    B. List
    C. Tuple
    D. Array

5. True or false: In Python, the array module is used for creating packed sequences.
    A. True
    B. False

6. What does the reshape() function do in NumPy?
    A. It returns an array with all values set to zero.
    B. It creates an array with a range of numbers.
    C. It solves a system of linear equations.
    D. It changes the shape of an existing array.

7. In Python, which module is used to perform accurate floating point calculations?
    A. decimal
    B. statistics
    C. fractions
    D. math

8. Which function from the math module would return the integer just above a floating-point number?
    A. math.floor()
    B. math.trunc()
    C. math.ceil()
    D. math.round()

9. Which function from the fractions module returns the greatest common divisor (GCD) of two numbers?
    A. gcd()
    B. common()
    C. lcm()
    D. factorial()

10. In NumPy, which method generates a 1D array of numbers from 2.0 to less than 10.0 with a step of 0.3?
    A. np.linspace(2.0, 10.0, 0.3)
    B. np.range(2.0, 10.0, 0.3)
    C. np.arange(2.0, 10.0, 0.3)
    D. np.sequence(2.0, 10.0, 0.3)

Day 14 to 16 – Summative Revision

Introduction:

Revision will be conducted to prepare students for their summative exams.

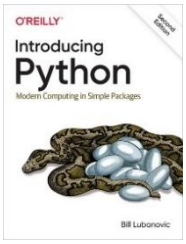
Day 17 – Summative Exam

Introduction:

Summative exams will be conducted on campus under invigilation. Strict exam rules will be enforced during the summative exam.

Practical for the practical modules will have to be handed in.

Bibliography

Textbook 1: Introducing Python by Bill Lubanovic. Available on O'Reilly Books online at: https://learning.oreilly.com/library/view/introducing-python-2nd/9781492051374/	
Textbook 2: Beginning T-SQL by Kathi Kellenberger and Scott Shaw. Available on O'Reilly Books Online at: https://learning.oreilly.com/library/view/beginning-t-sql/9781484200469/