COMP 1510 202210 Schedule

WEEK	DATE	MODULE	LECTURE 1 MONDAY	LECTURE 2 THURSDAY	LECTURE 3 THURSDAY	TYPES	BUILT IN FUNCTIONS	MODULES AND LIBRARIES	LAB DOMAIN	LAB TOPICS	1E LAB 1 MONDAY	1E LAB 2 MONDAY	1E LAB 3 MONDAY	1F LAB 1 TUESDAY	1F LAB 2 TUESDAY	1F LAB 3 THURSDAY	1E AND 1F TUTORIAL THURSDAY	1E AND 1F END OF TUTORIAL QUIZ	ASSIGNMENT TOPIC	ASSIGNMENTS	DUE DATES
1	Jan 3	Introduction	**NO CLASS**	Intro; **logistics**; communicating with Slack; selected **history** of programming; programming **language levels**; programming language **paradigms**	Introduction to **Python** a high-level multi-paradigm language; Python keywords; The **Zen** of Python and the notion of programming philosophies and methods; the **command line**; **operators**, operands, and **expressions**; operator precedence; commands like print(); **variables** and **assignment**	numbers and strings	print( )		**NO LAB**										Learning the COMP 1510 workflow: The Zen of Python! You will set up your toolchain -- Python3, pip3, PyCharm, git, GitHub, Hello world!	A1 released on Thursday	
2	Jan 10	Programming 101: computational thinking and flowcharts	Compound operators; basic input; data conversion and str(), int(), float(), etc.; type(); **data types** and why we need them: int, float, bool, and the rest (starting with str and None); () [] {} and <>	Text editors; IDEs and **PyCharm**; introduction to **version control** using git and GitHub	Four cornerstones of problem solving using **computational thinking**: decomposition, abstraction, pattern matching, algorithms; **structured code** using control statements for sequence, selection, repetition, and indirection; **flowcharts**	int, float, bool, str, None	type( ), input( ), float( ), int( ), isinstance( ), str( ), sum( )		The Zen of Python	Our toolchain aka getting started with programming and version control	Lab 1: Getting started with programming and version control. You will write and execute scripts in PyCharm using variables, assignment, commands, expressions, operators, operands, and types!	Lab 1 continued	Lab 1 conclusion	Lab 1: Write and execute scripts in PyCharm using variables, assignment, commands, expressions, operators, operands, and types	Lab 1 continued	Lab 1 conclusion	Variables, assignment, types, how I have set up my computer	Quiz 00: Getting to know the D2L quiz interface			A1 due on Friday
3	Jan 17	Sequence and selection, functions, indirection	Programming lifecycle; **sequence** in Python; truth value testing and **selection** (branching) in Python using **if**, if-else, and if-elif-else statements; **naming** conventions	**Strings** in detail; ASCII, Unicode, and code points; **mutability**; dot syntax; **formatting output** (f-strings, str.format, and %-formatting); built-in functions like len()	**Indirection and functions**; built-in functions; anatomy of the function; **argument passing semantics** (pass by value of reference); functional **decomposition**; first decomposition examples and techniques; the **main function**	function, method	len( ), abs( ), chr( ), dir( ), help( ), ord( ), pow( ), round( )	string, pydoc	Computational thinking	Programming 101: sequence and selection	Lab 2: Computational Thinking. You will generate and represent decomposed algorithmic solutions with flowcharts, and implement corresponding solutions that employ sequence and selection!	Lab 2 continued	Lab 2 conclusion	Lab 2: Represent decomposed algorithmic solutions with flowcharts and implement corresponding solutions that employ sequence and selection	Lab 2 continued	Lab 2 conclusion	PyCharm, git, CT, flowcharts, processing user input	Quiz 01: material from last week	Computational thinking and flowcharts	A2 released on Monday	
4	Jan 24	Documentation, memory model, Intro to data structures and repetition	**Documentation**; comments and docstrings for functions and modules including pre- and post-conditions and **doctests**; **scope** and the Python **memory model** (stack, heap, variables, references, addresses, objects, interning, identity is/id() vs equality ==)	Intro to **data structures** and containers; **lists**; working with lists	**Membership** operators in and not in; **slicing** lists; intro to **repetition** (looping) with the for-loop, the **range** function	list	list( ), any( ), id( ), filter( ), map( ), max ( ), min( ), sorted( )		Modularity, reusabililty, and encapsulation	Functions, indirection, and memory	Lab 3: Modularity and reusability. You will implement and document first functions using indirection, decomposition, selection, user input, built in-functions.	Lab 3 continued	Lab 3 conclusion	Lab 3: Implement and document first functions using indirection, decomposition, selection, user input, built in-functions.	Lab 3 continued	Lab 3 conclusion	Selection, mutability, functional decomposition, doctests	Quiz 02: material from last week			A2 due on Friday
5	Jan 31	Debugging and testing, standard library, identity vs equality	Lists and **identity vs equality**; copying (**deep vs shallow** copies); memory management and garbage collection	Tuples; passing functions to functions as objects	filter and map; intro to **debugging**; debugging with PyCharm	range, tuple	range( ), tuple( ), reversed( )	math, random, copy, pprint, statistics, getpass, builtins	Parsimony and clarity	Intro to data structures, repetition, and debugging	Lab 4: Parsimony and clarity. You will consider parsimony and clarity while implementing, documenting, testing and debugging a solution to a brain teaser that uses lists and looping, math module, random module, copy, pprint, etc.	Lab 4 continued	Lab 4 conclusion	Lab 4: Parsimony and clarity. You will consider parsimony and clarity while implementing, documenting, testing and debugging a solution to a brain teaser that uses lists and looping, math module, random module, copy, pprint, etc.	Lab 4 continued	Lab 4 conclusion	Debugging code (identity vs equality and deep vs shallow copies), lists, repetition	Quiz 03: material from last week	A tested module of independent functions	A3 released on Monday	
6	Feb 7	Testing, more data structures	Exploring the **standard library**; math module; **random** numbers; **constants**; built-in constants	**Errors**, synax and semantics; **testing**; disjointed equivalency partitions and coverage; automated testing; **unit testing**; assertions	More unit testing examples; **Boolean expressions** and, or, not; short-circuiting; floats and rounding	dictionary, iterable, iterator, generator	dict( ), zip( ), enumerate( ), filter( ), iter( ), next( ), set( )	unittest, itertools, doctest	Cryptography and ciphers	Testing, debugging, and more data structures	Lab 5: Cryptography and ciphers. You will build, test, and debug a small module of related atomic functions to help me encrypt and decrypt some ciphers.	Lab 5 continued	Lab 5 conclusion	Lab 5 : Cryptography and ciphers. You will build, test, and debug a small module of related atomic functions to help me encrypt and decrypt some ciphers.	Lab 5 continued	Lab 5 conclusion	Prepare for midterm	Quiz 04: material from last week			A3 due on Friday
7	Feb 14	**NO CLASS MIDTERM EXAMS**																			
8	Feb 21	Dictionaries, iteration, syntactic sugar, while loop	**NO CLASS**	**Dictionaries**; iteration, Iterables and **iterators**;	itertools and zip(); using enumerate() instead of range; ranges vs iterators vs views	set, module, package	set( )	unittest.mock, argparse, http, http. client, http.server	Data Communication	Echo client	Lab 6: Data communication. You will Implement and demonstrate an echo client! **NOTE THIS LAB IS AT-HOME BECAUSE MONDAY IS FAMILY DAY.**	Lab 6 continued **NOTE THIS LAB IS AT-HOME BECAUSE MONDAY IS FAMILY DAY.**	Lab 6 conclusion **NOTE THIS LAB IS AT-HOME BECAUSE MONDAY IS FAMILY DAY.**	Lab 6: Data communication. You will implement and demonstrate an echo client!	Lab 6 continued	Lab 6 conclusion	Doctests, unit tests, iteration	Quiz 05: material from last week	Lego Mindstorms Robots	A4 released on Monday	
9	Feb 28	Functions 2.0, decorators and closures	**Syntactic sugar** and list and dictionary **comprehensions**; nested data structures; repetition (looping) with **while**; sentinel values; breaking out of loops; **infinite** loops; loops and user input; **pass** statement	Syntactic sugar and conditional expressions; **sets**; more about **unit testing** (fixtures; mocking; generating input for tests; testing printed output; creating 'predictable' random numbers)	More about **functions**: default values; variable length parameter lists; positional and arbitrary arguments; keyword arguments; annotations; building good functions (implementing encapsulation, information hiding, message passing; decomposition; testing); simple recursion	decorator, closure		sys, time, typing, timeit	Programming style	Repetition, mocking	Lab 7: Programming style. You will build, document, annotate, test, and debug a small module of related atomic functions using dictionaries, iteration, nested data structures, and comprehensions.	Lab 7 continued	Lab 7 conclusion	Lab 7: Programming style. You will build, document, annotate, test, and debug a small module of related atomic functions using dictionaries, iteration, nested data structures, and comprehensions.	Lab 7 continued	Lab 7 conclusion	Repetition, nested data structures, mocking	Quiz 06: material from last week			A45 due during hour 3 of lab
10	Mar 7	Duck typing, exceptions, and file IO	Function decorators, inner functions, and closures	Compiling vs interpreting; **duck typing** (static vs dynamic) and strong vs weak typing; **sys.args** for command line arguments; passing command line arguments to the main function	**Exceptions**; try-except-else-finally; unit testing: testing for expected exceptions; exception hierarchy; commonly used exceptions; guard clauses are wasteful (LBYL vs EAFP)	exception	try-except, with		Profiling and optimizing	Function annotations, duck typing, modules and packages, functions 2.0	Lab 8: Profiling and optimizing. You will experiment with decorators and inner functions and consider the benefits and costs of LBYL vs EAFP.	Lab 8 continued	Lab 8 conclusion	Lab 8: Profiling and optimizing. You will experiment with decorators and inner functions and consider the benefits and costs of LBYL vs EAFP.	Lab 8 continued	Lab 8 conclusion	Typing, annotations, decorators, inner functions and closures	Quiz 07: material from last week	Text-based adventure game	A5 released on Monday	
11	Mar 14	**NO CLASS SPRING BREAK**																			
12	Mar 21	Refactoring, classes	**File IO**; opening, reading from, writing to, closing, deleting files; **context managers** and else blocks; context managers and file-like objects; working with **JSON**	**Modules and packages; refactoring**; code smells and the refactoring catalog: 1. the basics 2. encapsulation 3. moving features around 4. organizing data 5. clarifying logic 6. refactoring simple APIs	Intro to **classes**; attributes; **class-level variables**; instance initializers, validation and **invariants**; methods; classes vs objects; **state**	class, file-like objects, context manager	open( )	os, json, csv, zipfile, difflib, filecmp,os. path, secrets	Web scraping and simple APIs	Exceptions and file IO, refactoring	Lab 9: scrape date from the web and store an analysis in a file	Lab 9 continued	Lab 9 conclusion	Lab 9: Web scraping and simple APIs. You will work with files and data and learn how to scrape a webpage!	Lab 9 continued	Lab 9 conclusion	LBYL vs EAFP and exceptions	Quiz 08: material from last week			A5 due on Friday
13	Mar 28	Useful libraries and APIs	Designing good classes: **responsibility-driven design**, design before implementation, Abbot's heuristic; **visibility**, encapsulation, and information hiding; unit testing classes	Introducing Python libraries and APIs for desktop, web, and data science: keeping time, **scheduling** tasks, **launching** programs	Python library exploration: machine learning with **numpy** and **pandas** and **matplotlib**	array		datetime, subprocess, webbrowser, numpy, pandas, matplotlib	Flask and serving an API	Classes and APIs	Lab 10: Build, test, document, and publish a simple web API that uses classes and Flask	Lab 10 continued	Lab 10 conclusion	Lab 10: Flask and serving an API. You will build, test, document, and publish a simple web API using classes and Flask!	Lab 10 continued	Lab 10 conclusion	Refactoring, classes and how to use them	Quiz 09: material from last week	Web-based CRUD app	A6 released on Monday	
14	Apr 4	Useful libraries and APIs	Python library exploration: machine learning with **numpy** and **pandas** and **matplotlib**	Python library exploration: **regular expressions**	Python library exploration: **regular expressions**			re	Machine learning	Putting it all together	Lab 11: Implement and use a fundamental machine learning algorithm	Lab 11 continued	Lab 11 conclusion	Lab 11: Machine learning. You will use arrays, math, and Python to make some predictions!	Lab 11 continued	Lab 11 conclusion	Preparing for the final exam	Quiz 10: material from last week and this week			A6 due on Friday
15	Apr 11	**NO CLASS FINAL EXAMS**	Final review																		
16	Apr 18																				