COURSE OUTLINE: CSD105 - PYTHON

Prepared: Rodney Martin
Approved: Martha Irwin - Dean

Course Code: Title	CSD105: PYTHON
Program Number: Name	2090: COMPUTER PROGRAMMER 2095: COMPUTER PROGRAMMING 4029: ELECTRICAL TY-PROCES
Department:	COMPUTER STUDIES
Academic Year:	2025-2026
Course Description:	The Python programming language is a popular and easy-to-learn programming language that allows students to become comfortable with the fundamentals of programming without the troublesome syntax that can be challenging for novices. With the knowledge acquired using Python, students will have the knowledge to solve computational problems using the foundational concepts of all programming languages, namely: variables, basic data structures such as tuples, lists, and dictionaries, conditional and looping structures, and functions.
Total Credits:	3
Hours/Week:	3
Total Hours:	42
Prerequisites:	There are no pre-requisites for this course.
Corequisites:	There are no co-requisites for this course.
Substitutes:	CSD104, ELN331
This course is a pre-requisite for:	CSD102, CSD203, ELN340, GIS428, GIS440
Vocational Learning Outcomes (VLO's) addressed in this course: **Please refer to program web page for a complete listing of program outcomes where applicable.**	**2090 - COMPUTER PROGRAMMER** VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. VLO 10 Cntribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process. **2095 - COMPUTER PROGRAMMING** VLO 2 Contribute to the diagnostics, troubleshooting, documenting and monitoring of technical problems using appropriate methodologies and tools. VLO 10 Contribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development

	methodologies based on defined specifications and existing technologies/frameworks.
VLO 11	Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process.

4029 - ELECTRICAL TY-PROCES

VLO 2	Analyze and solve complex technical problems related to electrical systems by applying mathematics and science principles.
VLO 8	Use computer skills and tools to solve a range of electrical related problems.

Essential Employability Skills (EES) addressed in this course:

EES 3	Execute mathematical operations accurately.
EES 4	Apply a systematic approach to solve problems.
EES 5	Use a variety of thinking skills to anticipate and solve problems.
EES 10	Manage the use of time and other resources to complete projects.

Course Evaluation:

Passing Grade: 50%, D

A minimum program GPA of 2.0 or higher where program specific standards exist is required for graduation.

Other Course Evaluation & Assessment Requirements:

To successfully pass this course, the student must receive passing grades for both the Test portion of the class AND the Laboratory portion.

Grade
Definition Grade Point Equivalent
A+ 90 - 100% 4.00
A 80 - 89%
B 70 - 79% 3.00
C 60 - 69% 2.00
D 50 - 59% 1.00
F (Fail) 49% and below 0.00

CR (Credit) Credit for diploma requirements has been awarded.
S Satisfactory achievement in field /clinical placement or non-graded subject area.
U Unsatisfactory achievement in field/clinical placement or non-graded subject area.
X A temporary grade limited to situations with extenuating circumstances giving a student additional time to complete the requirements for a course.
NR Grade not reported to Registrar`s office.
W Student has withdrawn from the course without academic penalty.

Books and Required Resources:

Think Python: How to Think Like a Computer Scientist by Allen B. Downey
Publisher: O`Reilly Edition: 3
ISBN: 9781098155438
Freely available online: https://allendowney.github.io/ThinkPython/

Object-Oriented Programming in Python
Publisher: readthedocs.org
Freely available online: https://python-textbok.readthedocs.io/en/stable/

Course Outcomes and Learning Objectives:

Course Outcome 1	**Learning Objectives for Course Outcome 1**
Describe the nature of computers and programming	1.1 Define computation, and explain how it relates computers and programming languages 1.2 Explain what a programming language is (and is not) 1.3 Distinguish between compiled and interpreted languages 1.4 Explain what is meant by a language`s syntax 1.5 Describe what happens in a computer when you run a program 1.6 Describe the basic elements of all computer programs 1.7 Use a read-evaluate-print loop (REPL) to execute instructions and experiment with ideas 1.8 Use a text editor and interpreter to create programs
Course Outcome 2	**Learning Objectives for Course Outcome 2**
Create variables and simple expressions and statements	2.1 Define the terms `value` and `type` 2.2 Determine the type of a value, and cast values from one type to another 2.3 Create values of various types, including integers, floating point numbers, and strings 2.4 Explain the use of null values 2.5 Use values, operators and operands to create expressions 2.6 Explain operator precedence 2.7 Create useful code comments 2.8 Assign values to variables, and describe how this looks at the level of computer memory 2.9 Describe variable naming conventions 2.10 Distinguish between expressions and statements
Course Outcome 3	**Learning Objectives for Course Outcome 3**
Use and create functions	3.1 Describe what a function is and why it is useful 3.2 Identify when to encapsulate instructions into a function 3.3 Create functions that may include parameters and/or yield values 3.4 Call functions using arguments 3.5 Use function results as values in expressions that may include function composition 3.6 Analyze flow of program execution when functions are involved 3.7 Discuss variable scope 3.8 Provide function documentation using conventional syntax 3.9 Explain function preconditions and postconditions
Course Outcome 4	**Learning Objectives for Course Outcome 4**
Control program flow using conditionals	4.1 Create boolean expressions using relational and logical operators 4.2 Explain when and how non-boolean values may be interpreted as boolean values 4.3 Use conditional statements to control program flow, including chained and nested conditionals 4.4 Use conditional statements to check function preconditions 4.5 Describe the limitations of equality comparisons with

	floating point numbers
Course Outcome 5	**Learning Objectives for Course Outcome 5**
Control program flow using loops	5.1 Use loops to repeat a set of instructions a fixed number of times 5.2 Use loops to repeat instructions depending on a dynamic condition 5.3 Describe and use counter variables and sentinel values 5.4 Explain infinite loops and understand how to avoid them 5.5 Use return, break and continue statements to end loops early
Course Outcome 6	**Learning Objectives for Course Outcome 6**
Use objects	6.1 Explain what an \`object\` is 6.2 Describe object methods and properties 6.3 Use the methods and properties of objects in working code 6.4 Distinguish between mutable and immutable types 6.5 Explain what a reference is, and describe the underlying model in terms of computer memory 6.6 Discuss the difference between object equality and object identity 6.7 Distinguish between aliasing assignment and object cloning
Course Outcome 7	**Learning Objectives for Course Outcome 7**
Use sequences (lists, tuples, strings) and dictionaries to store and track information	7.1 Describe what a data structure is 7.2 Explain what a sequence is, and how it pertains to lists, tuples, and strings 7.3 Obtain one element of a sequence using indexing 7.4 Obtain subsections of a sequence using slicing 7.5 Determine if an element is in a sequence 7.6 Remove an element from a sequence 7.7 Traverse sequences using loops 7.8 Create strings to represent textual data 7.9 Analyze list/tuple/string objects using common methods 7.10 Create and use nested lists and/or tuples and understand when they are useful 7.11 Explain how dictionaries differ from sequences such as lists 7.12 Map a key to a value using a dictionary 7.13 Obtain a dictionary item using indexing 7.14 Determine if a dictionary key has already been set 7.15 Traverse dictionary data using loops 7.16 Create and use dictionaries of lists/tuples
Course Outcome 8	**Learning Objectives for Course Outcome 8**
Employ basic software design techniques	8.1 Describe the purpose of modules 8.2 Reuse existing code by importing modules 8.3 Describe the terms \`encapsulation\` and \`generalization\`, and how they pertain to functions and modules 8.4 Explain what is meant by a function or module\`s interface 8.5 Use refactoring to improve existing code 8.6 Explain what an algorithm is and be able to implement

		simple algorithms
	Course Outcome 9	**Learning Objectives for Course Outcome 9**
	Handle input/output, and errors	9.1 Produce output using a print statement 9.2 Employ string formatting techniques 9.3 Obtain keyboard input from a user 9.4 Distinguish between absolute and relative file paths 9.5 Create paths to specific files 9.6 Read data from a file 9.7 Write data to a file 9.8 Distinguish between syntax, runtime, and semantic errors 9.9 Prevent program crashes due to errors using try..catch blocks 9.10 Use debugging tools to investigate errors

Evaluation Process and Grading System:

Evaluation Type	Evaluation Weight
Lab Assignments	40%
Quizzes	10%
Test 1	25%
Test 2	25%

Date:	June 19, 2025
Addendum:	Please refer to the course outline addendum on the Learning Management System for further information.