## Course Description:
This course is intended for students majoring in Computer Engineering Technology, Electronics Engineering Technology, or any engineering discipline. Students will learn how to use object oriented programming to analyze, design and code programs to solve engineering related hardware problems. (3-hour lecture, 2-hour lab)

### Course Competency | Learning Outcomes
--- | ---
**Competency 1:** The student will demonstrate knowledge of basic computing and problem solving by:
1. Understanding the software development life cycle: analyze, design, code, test/correct, and document the program.
2. Developing programming algorithms.
3. Utilizing tools to convert algorithms into computer programs including flowcharts, pseudocode, hierarchy charts, and decision structures.
4. Demonstrating the use of the text editor IDLE.
5. Demonstrating the use of the python shell.
6. Learning to use the python code editor.
   - Critical thinking
   - Information Literacy

**Competency 2:** The student will demonstrate knowledge of core objects, variables, input and output by:
1. Demonstrating a knowledge of python’s core objects including integers, floats, strings, lists, tuples, files, sets, and dictionaries.
2. Demonstrating an understanding of variables.
3. Utilizing labels in arithmetic operations.
4. Demonstrating an ability to read and display data using the input and print functions.
   - Critical thinking
   - Information Literacy

**Competency 3:** The student will demonstrate an understanding of structures that control program flow by:
   - Critical thinking
   - Information Literacy

---
Updated Spring 2021
1. Demonstrating the use of relational and logical operations.
2. Demonstrating the use of decision structures: if, if-else.
3. Demonstrating the use of the while and for loops.
4. Demonstrating the use of the break and continue statements.

**Competency 4:** The student will demonstrate an understanding of functions by:

1. Demonstrating the use of built-in functions and user-defined functions.
2. Understanding the scope of variables and constants.
3. Creating library modules.
4. Demonstrating the ability to use lambda expressions.
5. Learning how to use functions from the random module.
6. Demonstrating the use of recursive functions.
7. Applying top-down design in problem solving.

**Competency 5:** The student will understand how to process data in permanent storage by:

- Critical thinking
- Information Literacy

1. Demonstrating an ability to read and create text files.
2. Demonstrating how to add lines/items to a text file.
3. Explaining how to use set-theoretic methods with text files.
4. Demonstrating how to read and create csv files.
5. Demonstrating how to analyze data in a csv file with a list.
6. Learning to use dictionaries with files.

**Competency 6:** The student will understand the basic concepts of object oriented programming by:

- Critical thinking
- Information Literacy

Updated Spring 2021
1. Defining the difference between built-in classes and user-defined classes.
2. Building user-defined classes.
3. Defining inheritance by new classes that are modified versions of existing classes.
4. Comparing polymorphism by creating methods with the same header but different definitions.
5. Creating a new instance variables to a subclass.
6. Demonstrating the ability to override a method.

**Course Competency 7:** The student will understand the difference between text-based user interface and a graphical user interface by:

- Critical thinking
- Information Literacy

1. Implementing programs that utilize buttons, labels, entry boxes and scroll bars.
2. Utilizing one of the three geometry managers (grid, pack, and place) to place widgets on the computer screen.
3. Designing screen layouts.