El Camino College
COURSE OUTLINE OF RECORD – Approved

I. GENERAL COURSE INFORMATION

Subject and Number: **Computer Science 14**
Descriptive Title: **Introduction to Programming with Python**
Course Disciplines: **Computer Science**
Division: **Mathematical Sciences**

Catalog Description:
This course is an introduction to computer programming and algorithm design using Python programming language. The course covers the fundamentals of Python programming: basic data types, objects, Switching and looping constructs, functions, recursion, lists, dictionaries and basic input and output, both interactive and with files.

Conditions of Enrollment:
Prerequisite: Mathematics 170 with a minimum grade of C or qualification by appropriate assessment or equivalent skill

Course Length:	**X Full Term**	**Other (Specify number of weeks):**
Hours Lecture:	**3.00 hours per week**	**TBA**
Hours Laboratory:	**3.00 hours per week**	**TBA**
Course Units:	**4.00**	

Grading Method: Letter
Credit Status: **Associate Degree Credit**

Transfer CSU:	**X Effective Date: 1/20/2016**
Transfer UC:	**X Effective Date: Proposed**

General Education:

El Camino College:
4B – Language and Rationality – Communication and Analytical Thinking
Term: Fall 2016 Other:

CSU GE: B4. Mathematics/Quantitative Reasoning and B3

IGETC: Area 2 Mathematical Concepts and Quantitative Reasoning

II. OUTCOMES AND OBJECTIVES

A. COURSE STUDENT LEARNING OUTCOMES (The course student learning outcomes are listed below, along with a representative assessment method for each. Student learning outcomes are not subject to review, revision or approval by the College Curriculum Committee)

1. Writing algorithms: Students write and correct detailed algorithms, some of them would include text processing. (Properly analyze a software problem using top down design, and write related algorithm that can be translated into computer program in Python).

2. Students will write Python code that uses correct syntax (when using data types, writing algebraic and logical expressions, naming variables, etc.)

3. Control Structures: Students will write Python code that correctly uses control structures (and nested control
structures) including conditionals (like "if"), loops (like "while" and "for") and user defined functions (both void and value returning).

4. Basic Data Structures: Students will write Python code that correctly uses basic data structures (including strings, lists, dictionaries, and intro to classes).

The above SLOs were the most recent available SLOs at the time of course review. For the most current SLO statements, visit the El Camino College SLO webpage at http://www.elcamino.edu/academics/slo/.

B. Course Student Learning Objectives (The major learning objective for students enrolled in this course are listed below, along with a representative assessment method for each)

1. Explain Computer Programming concepts.
   - Objective Exams
2. Convert algorithms to Python programs.
   - Laboratory reports
3. Design modular Python programs using functions.
   - Laboratory reports
4. Design programs with Interactive Input and Output
   - Laboratory reports
5. Ability to design programs utilizing arithmetic expressions.
   - Other exams
6. Design programs utilizing repetition.
   - Laboratory reports
7. Design programs utilizing decision making.
   - Performance exams
8. Design programs utilizing lists, dictionaries, and strings.
   - Other exams
9. Design programs using file Input and Output.
   - Laboratory reports
10. Develop simple search and sort algorithms.
    - Other exams
11. Introduction to Object Oriented Programming (OOP).
    - Other (specify)
    - Lab assignment, to complete a partially written program using OOP techniques.

III. OUTLINE OF SUBJECT MATTER (Topics are detailed enough to enable a qualified instructor to determine the major areas that should be covered as well as ensure consistency from instructor to instructor and semester to semester.)

Lecture or Lab	Approximate Hours	Topic Number	Major Topic
Lecture	4	I	Fundamentals of the Python language for computer science programming A. Use of the computer and computer languages B. Problem analysis C. Designing algorithm
Lecture	4.5	II	Elementary data types and operations A. Data types 1. Numeric 2. String B. Operations 1. Assignment operator 2. Arithmetical operations
Lecture	9	III	Design with control structures A. Design with decision making steps: use of if, if-else, nested if, multi- alternative if, and switch statements B. Design with repetitions steps: use of iteration statements C. Pre-test and post-test loops and software situations identifying their use. D. Introduction to End-of-file control loops and file processing based on them.
Lecture	12	IV	Design with sub programs (block-structured programming style) A. Functions with and without parameters/return values B. Use of control structures in the context of sub programs C. Keyword arguments and argument unpacking D. Built in functions in Python E. Text processing with built-in and user defined functions in Python. F. Functions returning multiple parameters G. Functions returning tuples
Lecture	3.5	V	Errors and Exceptions A. General introductions to errors and exceptions B. Raising and handling exceptions C. User Defined exceptions
Lecture	7.5	VI	Input/Output File manipulations A. Use of text input files B. creating output files C. Output formatting D. Methods of File object
Lecture	10.5	VII	Design with large block of data in main memory A. Use of structured data-types:

			1. Set 2. Dictionary 3. List 4. Tuple B. Operations 1. Read 2. Print 3. Search 4. Sort 5. Modify
Lecture	3	VIII	Classes and objects A. Demonstrating design and development B. Use of a user-defined class
Lab	4	IX	Fundamentals of the Python language for computer science programming A. Use of the computer and computer languages B. Problem analysis C. Meaning of an algorithm
Lab	4.5	X	Elementary data types and operations A. Data types 1. Numeric 2. String B. Operations 1. Assignment operators 2. Arithmetical operations
Lab	9	XI	Design with control structures A. Design with decision making steps: use of if, if-else, nested if, multi- alternative if, and switch statements B. Design with repetitions steps: use of iteration statements C. Pre-test and post-test loops and software situations identifying their use. D. Introduction to End-of-file control loops and file processing based on them.
Lab	12	XII	Design with sub programs (block-structured programming style) Functions with and without parameters/return values A. Use of control structures in the context of sub programs B. Keyword arguments and argument unpacking C. Built in functions in Python D. Text processing with built-in and user defined functions in Python. E. Functions returning multiple parameters F. Functions returning tuples
Lab	3.5	XIII	Errors and Exceptions A. General introductions to errors and exceptions B. Raising and handling exceptions

			C. User Defined exceptions
Lab	7.5	XIV	Input/Output File manipulations A. Use of text input files B. Creating output files C. Output formatting D. Methods of File object
Lab	10.5	XV	Design with large block of data in main memory A. Use of structured data-types: 1. Sets 2. Dictionaries 3. Lists 4. Tuples B. Operations 1. Read 2. Print 3.Search 4. Sort
Lab	3	XVI	Classes and objects A. Demonstrating design and development B. Use of a user-defined class

Total Lecture Hours	54
Total Laboratory Hours	54
Total Hours	108

IV. PRIMARY METHOD OF EVALUATION AND SAMPLE ASSIGNMENTS

A. PRIMARY METHOD OF EVALUATION:

Problem solving demonstrations (computational or non-computational)

B. TYPICAL ASSIGNMENT USING PRIMARY METHOD OF EVALUATION:

I. Process a list of numbers:

Write a program that first gets a list of integers from a file. Then prompt user for two more integers representing lower and upper bounds of a range. Your program should output all integers from the list that are within that range (inclusive of the bounds).

Ex: If the input is:

25 51 0 200 33 (list of numbers read from the file)
Enter lower bound of the range: 0
Enter upper bound of the range: 50

The output is: 25 0 33
The bounds are 0-50, so 51 and 200 are out of range and thus not output.

Deliverables:

1. Algorithm for the entire software.
2. Source Code in Python.
3. The test data used to test your program.
4. A brief description of notable obstacles you overcame.
5. Suggestions for further improvement for the program.

C. COLLEGE-LEVEL CRITICAL THINKING ASSIGNMENTS:

1. In this assignment you will write two separate programs, so you must create two projects. Both will do the same thing, but they will use different tools. Both programs will request the user to input a line of text. The program must then analyze the text to determine whether or not it is a palindrome.

A palindrome reads the same forwards and backwards while ignoring punctuation and spaces. For example, these are all palindromes: abba abccccba radar Dad Madam, I'm Adam A man, a plan, a canal, Panama!!! The letter a is also a palindrome, as is any letter by itself. Finally, using this definition, the empty string is also considered a palindrome. A palindrome is NOT case sensitive. The expression does not need to make sense as with abccccba.

For the program, you must create a loop that allows the user to test several expressions until the user chooses to quit. However, it must allow for any kind of user response such as y, Y, yes or Yes for the response.
Functions are required. One function should test an expression to see if it is a palindrome- it should have the prototypes:
bool isPal(const string&) and
bool isPal(const char[]),
depending on which program you are doing. You should also have a function that removes punctuation from the expression- how you code this is up to you. I will give suggestions each day in class.
After finishing both programs, compare and contrast the two approaches in a short, but well-written paragraph.

2. You work as a programmer at company that produces math software for kids. Since your manager knows you're a python expert she asks you to write a program for her. The program should calculate the perimeter and area of any arbitrary triangle. You gladly accept realizing that you can write modular code to do the assignment.
   The user should be able to specify a triangle in any of three ways:
   1) By giving the length of all three sides, or
   2) By giving the lengths of two sides and the measure of the included angle, or
   3) By giving the measures of two angles and the length of the included side.
   Once the user has entered the input, the program should perform the proper operations to calculate the lengths of all three sides of the triangle (a, b and c), the measures of all three angles (A, B and C), the perimeter of the triangle, and the area of the triangle. As you see in the diagram, side a lies opposite the angle A, side b lies opposite the angle B, and side c lies opposite the angles C. You must use functions to implement this program.
   Write a report where you answer the following questions:
   a. What were some notable obstacles you overcame?
   b. Which one took the least amount of code to solve?
   c. Which one took the greatest amount of code to solve?
   d. Why might one's approach required more code than the other?
   e. What are some of the strengths and weaknesses inherent in offering a user multiple input options?

D. OTHER TYPICAL ASSESSMENT AND EVALUATION METHODS:

Essay exams
Quizzes
Laboratory reports
Homework Problems
Multiple Choice
True/False

V. INSTRUCTIONAL METHODS

Laboratory
Lecture

Note: In compliance with Board Policies 1600 and 3410, Title 5 California Code of Regulations, the Rehabilitation Act of 1973, and Sections 504 and 508 of the Americans with Disabilities Act, instruction delivery shall provide access, full inclusion, and effective communication for students with disabilities.

VI. WORK OUTSIDE OF CLASS

Study
Required reading
Problem solving activities
Other (specify)
Developing Python programs

Estimated Independent Study Hours per Week: 6 hours

VII. TEXTS AND MATERIALS

A. UP-TO-DATE REPRESENTATIVE TEXTBOOKS

Computer Programming in Python zyBooks.com, 2019.

B. ALTERNATIVE TEXTBOOKS

Tony Gaddis, Starting Out with Python 4th Edition

C. REQUIRED SUPPLEMENTARY READINGS

Python 3.0 documents and programming environment from website below:
https://www.python.org/doc/

D. OTHER REQUIRED MATERIALS

Flash drive 500 MB capacity.

VIII. CONDITIONS OF ENROLLMENT

A. Requisites (Course and Non-Course Prerequisites and Corequisites)

Requisites	Category and Justification
Course Prerequisite Mathematics-170 or	Computational/Communication Skills
Non-Course Prerequisite or	Many of the more advanced projects/assignments for this course require knowledge of trigonometry and the problem solving required to work with trigonometric concepts. A student enrolling in this course without this knowledge or these skills is highly unlikely to pass this course.

Non-Course Prerequisite	Many of the more advanced projects/assignments for this course require knowledge of trigonometry and the problem solving required to work with trigonometric concepts. A student enrolling in this course without this knowledge or these skills is highly unlikely to pass this course.

B. Requisite Skills

Requisite Skills
Familiarity with computers

C. Recommended Preparations (Course and Non-Course)

Recommended Preparation	Category and Justification

D. Recommended Skills

Recommended Skills
None.

E. Enrollment Limitations

Enrollment Limitations and Category	Enrollment Limitations Impact

Course created by Satish Singhal on 09/18/2015.

BOARD APPROVAL DATE: 01/20/2016

LAST BOARD APPROVAL DATE: 12/16/2019

Last Reviewed and/or Revised by: Massoud Ghyam **Date: 09/05/2019**