I. INTRODUCTION

A. This course presents an introduction to computer programming for solving a variety of problems. The problem-solving consists of applying structured techniques and representing of algorithms using design tools to include testing, evaluation, and documentation. This course is intended for non-computer science and non-computer engineering majors. Emphasis is provided on the fundamentals of design, development, testing, and documentation of computer programs. Problem-solving applying structured techniques and representation of algorithms using design tools. Includes design with structured techniques and algorithms using pseudo code and/or graphical representation.

B. This course serves as a required or elective course on various degree plans. Curriculum plans for degrees and certificates are listed in the current Central Texas College catalog.

C. The delivery method of this course may be traditional lecture/lab, blended lecture/lab, or online.

D. Prerequisites: None.

II. LEARNING OUTCOMES

Upon successful completion of this course, the student will be able to:

A. Identify the major concepts of structured programming. (C7, C15, C18, F1, F2)
B. Illustrate the general concepts of structured design. (C7, C15, C18, F1, F2)
C. Use design tools. (C7, C15, C18, C19, F1, F2)
D. Solve problems using logic techniques. (C1, C7, C8, C16, C18, F1, F3, F4, F8)
E. Produce documented algorithms. (C1, C7, C8, C16, C18, F1, F3, F8)
F. Design and develop algorithms to solve problems. (C1, C7, C8, C16, C18, F1, F3, F8)
G. Demonstrate a fundamental understanding of software development methodologies, such as modular design, pseudo code, flowcharting, and structure charts. (C1, C7, C8, C16, C18, F1, F3, F8)
H. Demonstrate appropriate design, coding, testing, debugging, and documenting of computer programs that implement problem specifications and requirements. (C1, C3, C5, C6, C8, C11, C16, C17, C18, C19, F1, F2, F3, F4, F5, F7, F9, F11, F13, F17)
I. Apply computer programming concepts to new problems or situations. (C1, C3, C5, C6, C8, C11, C16, C17, C18, C19, F1, F2, F3, F4, F5, F7, F8, F9, F11, F13, F17)

III. INSTRUCTIONAL MATERIAL

A. The instructional materials identified for this course are viewable through www.cted.edu/books

B. Lecture Classes also require at least one USB storage device. Online students may use cloud based storage.

IV. COURSE REQUIREMENTS

A. Attend both lecture and lab or in the case of online delivery, be actively engaged in Blackboard and maintain constant progress.

B. Be prepared to participate in discussion, team projects/assignments and take unannounced assessments relating to the lecture materials.

C. Complete all exams/assessments.

D. Submit all assignments on time.

V. ASSESSMENTS

A. Student content mastery will be evaluated in the following areas:
   • Assessments (midterm exam, quizzes, projects, discussion etc.)
   • Final Assessment (final exam and/or semester project, participation)

B. Scheduled and unscheduled assessments will be given at the discretion of the instructor.

C. Exams/assessments may be composed of both subjective and objective questions plus computer output.
D. A student must take all exams/assessments. Both online and on campus students who know in advance that they will be absent due to school sponsored trips, military duty or orders, or any other valid reason, must arrange to take an early exam/assessment. Unexpected absences due to illness or other extenuating circumstances will require the student to contact the instructor about make-up work in lieu of the missed exam/assessment.

E. Students with unexcused absences will be given a zero for each missed assignment.

VI. SEMESTER GRADE COMPUTATIONS

<table>
<thead>
<tr>
<th>Course Requirements</th>
<th>Points</th>
<th>Points</th>
<th>Grade</th>
<th>Quality Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td>300</td>
<td>900</td>
<td>A</td>
<td>Superior</td>
</tr>
<tr>
<td>Assessments</td>
<td>300</td>
<td>800</td>
<td>B</td>
<td>Above Average</td>
</tr>
<tr>
<td>Final Assessment</td>
<td>400</td>
<td>700</td>
<td>C</td>
<td>Average</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>1000</td>
<td>D</td>
<td>Passing but Un satisfactory</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F</td>
</tr>
</tbody>
</table>

VII. NOTES AND ADDITIONAL INSTRUCTIONS FROM THE INSTRUCTOR

A. Information on the following Academic Policies, as described in the CTC Course Catalog will be followed:
   1. Withdrawals
   2. Grading
   3. Class Attendance and Course Progress
   4. Scholastic Honesty

B. Cell Phones and Pagers: Students will silence cell phones and mobile devices while in the classroom or lab.

C. Americans with Disabilities Act (ADA): Disability Support Services provide services to students who have appropriate documentation of a disability. Students requiring accommodations for class are responsible for contacting the Office of Disability Support Services (DSS) located on the central campus. This service is available to all students, regardless of location. Review the website at [www.ctcd.edu/disability-support](http://www.ctcd.edu/disability-support) for further information. Reasonable accommodations will be given in accordance with the federal and state laws through the DSS office.
D. **Instructor Discretion:** The instructor reserves the right of final decision in course requirements and may make changes to the course outline and/or assignments as needed.

E. **Civility:** Individuals are expected to be aware of what a constructive educational experience is and be respectful of those participating in a learning environment. Failure to do so can result in disciplinary action up to and including expulsion.

VIII. COURSE OUTLINE

A. **Lesson One:** Introduction to Computers and Programming. Input, Processing and Output

1. **Learning Outcomes:** Upon successful completion of this unit the student will be able to:
   a. Discuss course requirements as defined in the syllabus and reviewed by the instructor
   b. Discuss the evolution of computing devices.
   c. Discuss general computer concepts.
   d. Discuss how a program works.
   e. Name and use constants, variables, and specific data types in a program.
      Use basic arithmetic operations (addition, subtraction, multiplication, division, modulus, and exponentiation) in a program.
   f. Describe the basic data types, and operations available with these types.
   g. Incorporate input, processing, and output into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. **Learning Activities:**
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. **Unit Outline:** Follow the learning activities

B. **Lesson Two:** Modular Design

1. **Learning Outcomes:** Upon successful completion of this unit the student will be able to:
   a. Describe the advantages of modular design.
   b. List the steps involved when defining and calling a module.
c. Describe the purpose of program design tools, such as a flowchart and hierarchy chart.
d. List the purpose, use and restrictions of local variables and local constants.
e. Discuss the terms argument and parameter their role in invoking modules.
f. Construct flowcharts and hierarchy charts.
g. List the purpose, use and restrictions of global variables and global constants.
h. Incorporate modular design structures into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. Learning Activities:
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. Unit Outline: Follow the learning activities

C. Lesson Three: Decision Structures and Boolean Logic

1. Learning Outcomes: Upon successful completion of this unit the student will be able to:
   a. Describe the logical purpose for decision structures.
   b. Write the program, using logical operators in pseudocode, a flowchart and in the Python language.
   c. Employ and implement dual and multiple alternative decision structures
      Identify and apply relational and logical operators in program segments.
   d. Implement the relational operators <, !=, ==, <=, >, and >= with strings.
   e. Construct nested selection structures.
   f. Write programs and flowcharts which employ both the if-elseif and case programming structure to execute multiway branching.
   g. Create complex conditional log structure using logical operators OR, AND, and NOT.
   h. List the purpose, use and restrictions of Boolean variables.
   i. Incorporate decision structures into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. Learning Activities:
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. **Unit Outline:** Follow the sequence of the unit objectives

D. **Lesson Four:** Repetition Structures

1. **Learning Outcomes:** Upon successful completion of this unit the student will be able to:
   a. Describe the purpose and syntax of repetition structures.
   b. Implement condition controlled loops using While, Do-While and Do-Until looping structures.
   c. Implement counter controlled loops using the For looping structure.
   d. Apply loops to compute sums and averages.
   e. Use counter-controlled loops to increment or decrement the counter by any integer value.
   f. Construct nested loops to achieve specified logic requirements.
   g. Incorporate repetition into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. **Learning Activities:**
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. **Unit Outline:** Follow the sequence of the unit objectives

E. **Lesson Five:** Functions

1. **Learning Outcomes:** Upon successful completion of this unit the student will be able to:
   a. Describe the overall purpose and advantages of using functions within a program.
   b. Incorporate the use of a library function which uses random numbers in a program.
   c. Develop, design and implement programmer defined functions.
   d. Discuss the meaning and use of return types and return values when designing and calling functions.
   e. List the range of function categories available from various programming libraries.
   f. Create, debug and test functions.
g. Incorporate library and user-defined functions into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. **Learning Activities:**
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. **Unit Outline:** Follow the sequence of the unit objectives

**F. Lesson Six: Input Validation**

1. **Learning Outcomes:** Upon successful completion of this unit the student will be able to:
   a. Describe the overall purpose and implementation of input validation.
   b. Implement an input validation loop using constraints identified in program requirements.
   c. Identify common input validation errors such as incorrect data type, out of range input, and boundary condition errors.
   d. Design and write programs which reflect sound defensive programming principles. Reflect these principles in flowcharts and in written code.
   e. Incorporate library and user-defined functions into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. **Learning Activities:**
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. **Unit Outline:** Follow the sequence of the unit objectives

**G. Lesson Seven: Arrays**

1. **Learning Outcomes:** Upon successful completion of this unit the student will be able to:
   a. Describe the overall purpose, implementation and structure of arrays.
   b. Discuss the advantages of using arrays as containers for large quantities of data.
c. Identify to use of subscripts or indices when using array notation to refer to array element position.
d. Design (using flowcharts) and write programs (Python) which use a loop as a basic construct to loop through an array and conduct processing on each array element.
e. Implement sound design and coding principles when using arrays, such as controlling for off by one errors, array bounds checking, and array initialization.
f. Design and write programs which search sequentially through an array for specified values, such as highest array value or lowest array value.
g. Implement functions which accept and process arrays received as a parameter.
h. Implement parallel arrays for sequences of data which are related by index position.
i. Design and write programs which use two-dimensional arrays to process data which more easily maps into a matrix structure.
j. Incorporate one and two-dimensional arrays into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.

2. Learning Activities:
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. Unit Outline: Follow the sequence of the unit objectives

H. Lesson Eight: Files

1. Learning Outcomes: Upon successful completion of this unit the student will be able to:
   a. Describe the purpose and advantages of storing and retrieving data from data files.
   b. Identify the types of data files, and the records and fields within a data file.
   c. Create, write data to, and read data from a sequential file.
   d. Incorporate the use of loop constructs to iterate through data files and perform operations on data elements within the file.
   e. Implement arrays to efficiently receive, temporarily store, and process data elements from a file.
   f. Incorporate file operations into the appropriate design, coding, testing, debugging, and documenting of Python computer programs which implement problem stated specifications and requirements.
2. **Learning Activities:**
   a. Research and discuss the topics of the Lesson in class and in an online collaborative discussion forum (C7, C8, C9, C15, C18, F1, F9, F10, F13)
   b. Perform skills and functions in the section (C1, C5, C7, C8, C9, C16, C18, F1, F9, F10, F13)

3. **Unit Outline:** Follow the sequence of the unit objectives