SOUTHEAST COMMUNITY COLLEGE
DIVISION OF ARTS AND SCIENCES
Mathematics
Revision Date: 07-01-22
Syllabus Statements

I. CATALOG DESCRIPTION

Course Number:	CSCI 1550
Course Title	Computer Science I
Prerequisite(s):	MATH1150 and MATH 1200 or MATH1300 or higher or equivalent test (Accuplacer) score
Catalog Description:	Introduction to problem solving with computers. Topics include problem solving methods, software development principles, computer programming, and computing in society.
Credit Hours:	4
Class Hours:	60
Lab Hours:	0
Total Contact Hours**:**	60

II. COURSE OBJECTIVES:

Mastery of the fundamentals of programming in a high-level language, including data types and rudimentary data structures, control flow, repetition, selection, input/output, and procedures and functions. Familiarity with problem solving methods, including problem analysis, requirements and specifications, design, decomposition and step-wise refinement, and algorithm development (including recursion). Familiarity with software development principles and practices, including data and operation abstraction, encapsulation, modularity, reuse, prototyping, iterative development, exception handling, documentation, coding conventions, and testing. Exposure to computing topics, including algorithms for searching and other problems, graphical user interfaces, event-driven programming, and database access. Exposure to the history of computing.

III. STUDENT LEARNING OUTCOMES AND GENERAL EDUCATION LEARNING OUTCOMES:

A. Student Learning Outcomes: *Student will be able to*:

This course has several learning objectives and "skills objectives." These are the skills that, upon successful completion of this course, you should be able to exhibit.

1. You should have a mastery of the fundamentals of programming in a high-level language, including data types and rudimentary data structures, control ow, repetition, selection, input/output, and procedures and functions.

2. You should be able to approach a reasonably complex problem, design a top-down solution, and code a program in a high-level programming language that automates solutions.

3. You should have a familiarity with problem solving methods, including problem analysis, requirements and specifications, design, decomposition and step-wise refinement, and algorithm development (including recursion).

4. You should have a familiarity with software development principles and practices, including data and operation abstraction, encapsulation, modularity, code and artifact reuse, prototyping, iterative development, best practices in coding design, style, and documentation, a good understanding of proper testing and debugging techniques and a familiarity with development tools.

5. You should have exposure to algorithms for searching, sorting and other problems, graphical user interfaces, event-driven programming, and database access.

6. You should have a foundation for further software development and exploration.

7. You should have a deep enough understanding of at least one high-level programming language that you should be able to learn another programming language with relative ease in a relatively short amount of time.

B. General Education Learning Outcomes

1. GELO #3: Critical Thinking & Problem Solving
Outcome: Collect, identify, interpret and analyze data.

2. GELO #5: Analytical, Quantitative, and Scientific Reasoning
Outcome: Apply mathematical and scientific methods to solve problems from an array of contexts and everyday situations.
Outcome: Understand and create logical arguments supported by quantitative and scientific evidence and communicate those arguments in a variety of formats.
Outcome: Effectively develop strategies, algorithms, or experiments (or performing experiments) to better describe the systems or to solve the problems.

IV. CONTENT/TOPICAL OUTLINE (*course outline may provide more detailed information)*

These topics may vary slightly to support and enhance achievement of the course objectives.

A. Writing Programs, Execution, and Interactive Design Environment (IDE)
B. Flow charts, Pseudocode
C. Data Types, Variables, Expressions, Input/Output
D. Tracing, Testing, and Debugging
E. Control Statements
F. Applications with Control Statements
G. Applications with Builtin Methods
H. Applications with Arrays
I. Encapsulating Data and Operations
J. Applications with Encapsulation
K. Modular Design and Programming
L. Evaluating Search and Sort Performance
M. Recursion
N. GUIs
O. Databases

V. INSTRUCTIONAL MATERIALS

A. Required Text(s):
1. Zelle, John, *Python Programming: An Introduction to Computer Science*, Franklin, Beedle & Associates Inc., 3rd edition, 2017. ISBN: 9781590282755.

B. Other Resources:
1. None.

VI. METHODS OF PRESENTATION/INSTRUCTION

A. Methods of presentation typically include a combination of the following:
1. Lecture
2. Lab
3. Group work

VII. METHODS OF EVALUATION

A. Methods of evaluation typically include a combination of the following:
1. Lab work
2. Homework
3. Mid-term exam
4. Final Exam

B. SCC STANDARD GRADING SCALE POLICY

A+	95-100	C+	75-79	F	59 or less
A	90-94	C	70-74		
B+	85-89	D+	65-69		
B	80-84	D	60-64		

VIII. SPECIFIC COURSE REQUIREMENTS

A. None