HOUSATONIC COMMUNITY COLLEGE

**Course Name:** Java Programming I

**Course Number:** CSC* E223

**Credits:** 4

**Catalog description:** An introduction to computer science and fundamentals of object-oriented programming using the Java programming language. The course emphasizes problem solving, algorithm development, and sound programming practices; the design, coding, testing and debugging, and documentation of computer programs. Students will learn how to construct algorithms and convert them into computer programs using typical flow control statements and data types, will learn basic object-oriented techniques and contrast them with function-oriented techniques, will work with primitive, compound and user-defined data types, will learn how to define functions and pass parameters to them, will use simple console input and output and simple file input and output, and will learn fundamental software engineering testing techniques. Topics include data types and expressions, packages, classes and objects, encapsulation, methods, conditionals, loops, arrays, and the Java API. The coursework includes a significant amount of homework assignments.

*The course requires substantial hands-on use of computers in a computerized classroom environment.*

**Prerequisite:** MAT* E137 or higher

**Corequisite or Parallel:**

## General Education Competencies Satisfied:

**HCC General Education Requirement Designated Competency Attribute Code(s):**

**None**

**Additional CSCU General Education Requirements for CSCU Transfer Degree Programs:**

**None**

**Embedded Competency(ies):**

**None**

**Discipline-Specific Attribute Code(s):**

**☒ COMP** **Computer Science**

## Course objectives:

**General Education Goals and Outcomes:**

**None**

**Course Specific Objectives:**

1. Learn the basic programming constructs, namely primitive data, variables, assignment statement, arithmetic operators, parameter passing

2. Evaluate and apply arithmetic, string and Boolean expressions

3. Learn the main control structures, namely: sequence, selection and loop. Apply nested if statements and loops

4. Understand and implement classes and use them in definite application settings. Apply the ideas of abstraction and encapsulation into program design

5. Understand and utilize text files and basic input/output operations

6. Learn arrays of primitive data and be introduced to arrays of objects

7. Gain experience with problem analysis interpreting non-trivial application problems and translating them into program design before implementing them in a programming language such as Java

8. Develop correct algorithms to solve a given problem and test and debug all aspects of the implements algorithm

**Course Content:**

1. Types, variables, using objects

   1.1 Objects and classes

   1.2 Data types and variables

   1.3 Calling methods; using accessor and mutator methods

   1.4 Constructing objects

   1.5 Using the API Documentation

   1.6 Object references

   1.7 Graphical applications: ellipses, lines, text, and color

2. Implementing Classes

   2.1 Instance variables and encapsulation

   2.2 Specifying the public interface of a class

   2.3 Providing the class implementation

   2.4 Local variables

   2.5 Data abstraction and encapsulation

   2.6 constructors and garbage collection

   2.7 The *this* reference

   2.8 Shape classes

3. Fundamental data types, decisions, loops

   3.1 Numbers

   3.2 Arithmetic

   3.3 Input and output

   3.4 Strings

   3.5 Boolean variables and operators

3.6 The *if* statement, comparing primitive values vs. comparing objects, nested branches

3.7 Simple input validation

3.8 Loops: *while*-, *for*-, *do*-, nested loops

3.9 Random number generation and simulations

3.10 Common loop algorithms: sum and other accumulation techniques, average, min, max, largest, working with sequences (including sequences requiring storage of more than one previous element, e.g. Fibonacci numbers and other similar sequences), finding the greatest common divisor, approximating the square root

4. Arrays and array lists

    4.1 Arrays, the enhanced *for* loop

    4.2 Common array algorithms

    4.3 Array lists

5. Basic validation, testing and debugging. Regression testing.

Date Course Created: Spring 2018

Date of Last Revision: 01/22/2018