Course Name: Java Programming II

Course Number: CSC* E224

Credits: 4

Catalog description: Continuation of CSC* E223 covering algorithm development, data structures and more advanced Java programming concepts. Topics include object-oriented design and class relationships, inheritance, polymorphism, Java interfaces, exceptions, models as abstractions of situations, simple simulation techniques, file input and output, introduction to event-driven programming, lists, stacks, queues, priority queues, sets, maps, binary search trees, sorting and searching, time complexity and space complexity, recursion, and Java Collections API. A substantial project component is included. Students must plan for sufficient time for out-of-class individual independent work.

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

Prerequisite: C or better in CSC* E223

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. Apply inheritance, polymorphism, Java interfaces and abstract classes

2. Design, implement, debug and test non-trivial Java applications (with multiple classes and/or interfaces with methods requiring two- and three-level nesting of loops and/or selection statements and/or requiring storage of two or more values form a previous iteration)

3. Design, implement, debug and test recursive algorithms

4. Write simple Java applets and simple GUI event-driven standalone applications

5. Understand models as abstractions of situations and describe simple modeling techniques

6. Write simple Java Graphics applications

7. Understand and utilize exception handling

8. Understand file input/output and streams; implement Java applications using file input/output and streams

9. Understand and utilize data structures

10. Understand Java collections; write Java applications using Java collections

11. Design recursive algorithms and implement recursive methods on integers, arrays, and Graphics

12. Apply, trace, and informally analyze searching and sorting algorithms

Course Content:

1. Review of Computer Science I topics. Introduction to modeling. Use of random numbers in simple simulations

2. Designing classes, inheritance

    2.1 Discovering classes in problem statements

    2.2 Guidelines for designing good methods

2.3 Static variables and methods

2.4 Packages

2.5 Unit test frameworks

2.6 Inheritance hierarchies

2.7 Implementing subclasses

2.8 Overriding methods

2.9 Polymorphism

2.10 Abstract classes

2.11 Object: the superclass of all classes

3. Graphics

3.1 Applets and their lifetime

3.2 Graphical applications and frame windows

3.3 Drawing on a component

3.4 Classes Graphics and Graphics2D

4. Interfaces

4.1 Using interfaces for algorithm Reuse

4.2 Working with interface variables

4.3 The Comparable interface

4.4 Using interfaces for callbacks

4.5 Inner classes

4.6 Event handling

4.7 Building applications with buttons

4.8 Processing timer events

4.9 Mouse events

5. Input/Output (extended review)

    5.1 Reading and writing text files

    5.2 Text input and output

    5.3 Command-line arguments

6. Object-Oriented Design

    6.1 Classes and their responsibilities

    6.2 Relationships between classes

7. Recursion (with plenty of examples and exercises!)

    7.1 Simple examples and exercises(e.g., number triangles, palindromes, etc.)

    7.2 Recursive helper methods

    7.3 The efficiency of recursion

    7.4 Non-trivial examples and exercises (e.g., permutations and other)

    7.5 Mutual recursion (example: interpreting arithmetic expressions)

    7.6 Backtracking

8. Introduction to Sorting and Searching

    8.1 Selection sort

    8.2 Informal analysis of the performance of the Selection Sort algorithm

    8.3 Searching: Linear vs. Binary Search

    8.4 Informally estimating the running time of an algorithm

9. Data structures

    9. 1 lists, stacks, queues, priority queues, sets, maps, binary search trees

10. Java collections

    10.1 Java Collections API

Date Course Created: Spring 2018

Date of Last Revision: 01/22/2018