North Hennepin Community College

CSCI 2001: Object Oriented Programming (CS1)

A. COURSE DESCRIPTION

Credits: 4

Lecture Hours/Week: 0

Lab Hours/Week: 0

OJT Hours/Week: *.*

Prerequisites:
This course requires both of these prerequisite categories
1. Any one of these three
   A score of 79 on test Accuplacer College Level Math
   A score of 3 on test Calculus Readiness Test
   MATH 1150 - College Algebra (Minimum grade: 1.67 GPA Equivalent)
   And
2. CSCI 1130 - Introduction to Programming in Java (CS0) (Minimum grade: 1.67 GPA Equivalent)

Corequisites: None

MnTC Goals: None

Students will learn object-oriented programming while creating algorithms.
The basic principles of software engineering are emphasized. By doing their own Java projects, students will develop
problem-solving skills and gain experience in detecting and correcting software errors.
Procedures, recursion, and iteration will be presented in the development of algorithms. Inheritance and polymorphism are studied. The use of abstraction will be emphasized throughout the course.

B. COURSE EFFECTIVE DATES: 07/02/2018 - Present

C. OUTLINE OF MAJOR CONTENT AREAS

1. Students will learn object-oriented programming while creating algorithms. They will write computer programs that are thoroughly documented and tested.
2. Topics include: Data types, Control Structures, Methods, Events, Arrays, Classes and Objects, Encapsulation, Inheritance
   Polymorphism, Interfaces, Re-usability

D. LEARNING OUTCOMES (General)

1. Apply consistent documentation and program style standards that contribute to the readability and maintainability of software. (ELO# 1,2)
2. Develop, design, analyze, and implement logic within a program that solves a problem with a finite number of operations. (ELO# 1,2)
3. Design and develop programs that implement fundamental logic structures of sequence, selection, and repetition.
4. Write programs that uses file I/O to provide persistence across multiple executions.
5. Develop proficiency in specification and use of appropriate primitive data types and their aggregation into simple linear data structures. (ELO# 1,2)
6. Decompose problems into clearly defined sub-problems based on program requirements. (ELO# 1,2)
7. Implement algorithms utilizing recursive structures. (ELO# 1,2)
8. Create appropriate test cases and use debugging skills to verify correctness of output. (ELO# 1,2)

E. Minnesota Transfer Curriculum Goal Area(s) and Competencies

None

F. LEARNER OUTCOMES ASSESSMENT

As noted on course syllabus

G. SPECIAL INFORMATION

None noted