Miami Dade College

GENERAL INFORMATION	
Course Prefix/Number: COP2654	Course Title: **Objective C Programming**
Number of Credits: 4 credits	
Degree Type	☐ B.A. ☐ B.S. ☐ B.A.S ☒ A.A. ☐ A.S. ☐ A.A.S. ☐ C.C.C. ☐ A.T.C. ☐ V.C.C
Date Submitted/Revised: 4/13/12	Effective Year/Term: 2012-1
☒ New Course Competency ☐ Revised Course Competency	
Course Description (limit to 50 words or less): This is an intermediate level programming course using the Objective C computer language, recommended for Computer Science and Computer Information Systems majors. Students will learn to code, compile and execute programs while learning advanced programming concepts and object oriented programming design concepts and principles. A.S. degree credit only. Special fee. (3 hr. lecture; 2 hr. lab)	
Prerequisite(s): COP2800	Corequisite(s):

Competencies:

Competency 1:
The student will demonstrate an understanding of the professional software development process by:

1. Designing and documenting solutions at the method level by writing pseudocode or developing flow charts for development before writing the code.
2. Designing and documenting solutions at the project level by using an object-oriented design technology such as UML or CRC cards.
3. Coding software solutions following professional coding style guidelines.
4. Incorporating adequate and meaningful comments into the source code.
5. Testing and designing tests of software solutions.
6. Debugging program code.

Competency 2:
The student will demonstrate an understanding of fundamental programming constructs and concepts by:

1. Using appropriate data types for programming assignments.
2. Using Boolean, comparison, arithmetic and object (instance of) operators in their programs.
3. Explaining the properties of a variable such as its name, value, scope, persistence, and size.
4. Distinguishing between expressions and statements.
5. Identifying and using the three control structures (sequence, selection and repetition).

Competency 3:
The student will demonstrate an understanding of the following advanced programming techniques by:

Revision Date:
Approved By Curriculum Report: 92 Reviewed By Director of Academic Programs Date: _____

1. Parsing a string and using other string manipulation techniques.
2. Using arrays to process aggregate data.
3. Using object composition (object references) to build more complex objects.

Competency 4:
The student will demonstrate an understanding of the object-oriented programming concepts of Class and Object by:

1. Identifying and using instance variables and instance methods.
2. Using, programming, and identifying constructors.
3. Explaining the process of object instantiation.
4. Using, programming, and identifying accessor and mutator methods.
5. Using, programming, and identifying class (static) variables and class (static) methods.
6. Using, programming, and identifying overloaded methods and constructors.

Competency 5:
The student will demonstrate an understanding of inheritance by:

1. Explaining the benefits of inheritance.
2. Creating a class which extends a parent class.
3. Explaining the restrictions imposed when using inheritance.
4. Overriding and overloading parent class functions within a child class.
5. Distinguishing between inheritance of implementation (extends) and inheritance of design (implements).
6. Creating a class which implements an interface.
7. Creating a class which extends an abstract class.

Competency 6:
The student will demonstrate an understanding of object-oriented design concepts by:

1. Using visibility modifiers (public, private, protected) to implement appropriate abstraction and encapsulation.
2. Explaining coupling and how to achieve loose coupling.
3. Explaining cohesion and how to achieve high cohesion.
4. Writing a program which demonstrates polymorphism.

Competency 7:
The student will demonstrate an understanding of Java input and output (I/O) by:

1. Describing I/O.
2. Creating programs that use console I/O.
3. Creating programs that use file I/O.

Competency 8:
The student will demonstrate an understanding of exception programming techniques by:

1. Describing exceptions.
2. Encapsulating exceptions.
3. Throwing and catching exceptions.