Linfield College, Computer Science Department
COMP 375: Interactive Application Design
Syllabus Spring 2017
Prof. Daniel Ford

COURSE DESCRIPTION:
COMP 375 introduces the fundamental principles and techniques of interactive application design. This class will involve extensive hands-on development of interactive iOS 10 applications using the Swift 3.1 programming language. Lecture/discussion. 3 credits.

PREREQUISITES:
Completion of COMP 161 or instructor approval is required.

LECTURES: Renshaw CMLB
Section 1: MWF 11:00 – 11:50 AM

PROFESSOR: Daniel Ford
Office: Renshaw 210 Phone: (503) 857-2995 E-mail: dford@linfield.edu
Office Hours: MWF 12:00 – 1:00 PM, TTh 12:15 – 1:15 PM, and by appointment.

COURSE OBJECTIVES:
- Understand and use the Model-View-Controller pattern.
- Explain and use GUI programming concepts.
- Design, evaluate, and implement interfaces and input/output alternatives.
- Facility with functional and event-driven programming.

Categories:

	Core	Advanced
Data Structures		
Algorithms		
Software Design		2.0
Computer Architecture		
Programming Languages		1.0

COURSE RESOURCES:

ITUNES U COURSE
- Stanford's Developing iOS 9 Apps with Swift (CS193p Spring 2016); Paul Hegarty.

REFERENCE BOOK
- iOS Programming: The Big Nerd Ranch Guide (6th Edition – January 2017); Christian Keur and Auron Hillegass; 978-0134682334.

SWIFT PROGRAMMING GUIDES & SAMPLE CODE
- The Swift Programming Language (Swift 3.1)
- Swift Standard Library.playground
- Swift Standard Library (API) Reference

GRADES:
Grading for this course will be based on:
Projects ~64%
Exams ~36%

COMMENTS
Most, if not all, of your notes for this class should be comments in one or more programs. It is very important that explain every important detail about every line of code you write in a comment, both to remind yourself later on and to show me that you understand what your code does and why you chose to write it that way.

DEBUGGING
A major objective of this courses is practice debugging, i.e. generating and implementing ideas to investigate and determine the cause of problems, a.k.a. bugs, as well as ideas to generate, test, evaluate and select solutions to fix these problems. These debugging ideas are implicit in any solution you submit for credit.
The purpose of having credit for the course based on problem solving and debugging skills is that repeated practice in generating and evaluating solutions is the best way to increase your programming and problem solving skills.

BE SMART ASK WHY
Being smart requires problem solving. Problem solving requires learning. Learning requires asking why. Whenever tempted to ask what you should do, try asking why something is happening instead.

TEST YOUR ASSUMPTIONS
We answer questions either by asking someone, looking something up, e.g. via a book, API, google search, stackoverflow.com , or the scientific method, i.e. testing hypotheses. Of these, testing hypotheses is the most important and requires the most practice. Running a program is akin to running an experiment. Running frequent small programs/experiments that differ from another test in only one way helps us learn much more and learn much more easily than using a few, infrequent, large, radically different tests.

Virtual Private Network (VPN)
The CS department has a VPN server that you can use to access all the files in your Z drive in the lab from anywhere in the world so that you can have all your work in one place. In addition all these files are backed up automatically. Tutorials on how to set this up are available in the content section of this class on blackboard.

ACADEMIC CONDUCT AND GUIDELINES
In this course we will adhere to the college policy on academic honesty, as published in the Linfield College Course Catalog. Please review this policy if you are not already familiar with it.
Plagiarism is attempting to get credit for someone else's work.

- Never copy, by any means, code that you did not create yourself into either a final or preliminary draft of an assignment that you will turn in for credit unless you are explicitly instructed to do so by the instructor.
    - Obey the *30 second long-term memory rule*. Take a minute to think about someone else's solution why it works or doesn't work and how you might improve on their idea before starting to write your own solution from scratch. If your implementation doesn't work see if you can fix it on your own first, and then repeat this step as necessary.
- Never turn in code you do not completely understand, and always use comments to help explain what you did and why you did it that way. The more complicated and difficult your code is to understand, the more comments you need to include with it to explain how it works and why you made the decisions you did.

In computer science you will earn credit by combining basic building blocks, e.g. the vocabulary of a specific language, common patterns, well known data structures, and algorithms, into expressions, segments of code, procedures, functions, and entire programs that solve a particular problem or accomplish a specific task. The basic building blocks are common knowledge and thus represent ideas that do not need to be cited. You are expected to generate, evaluate, and illustrate your own ideas on how these building blocks can be put together to solve particular problems. The building block combinations that you submit for credit will illustrate a combination of ideas that is similar to the combinations used by others while remaining uniquely your own.

ILLNESS/MEDICAL POLICY
In the unfortunate event that you must miss class you are encouraged to contact your workshop partner to find out what we worked on in class and to pick up any materials that were handed out. In general, homework extensions and make-up exams will be granted as needed on an individual basis.

DISABILITY SUPPORT:
Students with disabilities are protected by the Americans with Disabilities Act and Section 504 of the Rehabilitation Act. If you are a student with a disability and feel you may require academic accommodations contact Cheri White, Assistant Director of Learning Support Services (LSS), within the first two weeks of the semester to request accommodations. LSS is located in Walker 126 (503-883-2444). We also recommend students communicate with their faculty about their accommodations and any special needs an instructor should be aware of.