1. **Course number and name:** CSCI 1302 Programming Principles II

2. **Credits and contact hours:** 3 credit, 3 contact

3. **Instructor’s or course coordinator’s name:** Y Daniel Liang, PhD/ James Harris, PhD

   a. **Other supplemental materials:** None

5. **Specific course information**
   a. **Brief description of the content of the course (Catalog Description)**
      Emphasis is on advanced techniques such as recursion, regular expressions, refactoring, object oriented programming concepts and constructs, reusing components, templates/generics, interfaces and classes. Experiences include use of an integrated development environment and shared (code) repositories.
   b. **Prerequisites:** A minimum grade of “C” in CSCI 1301, MATH 1441, and MATH 2130.
   c. **Indicate whether a required, elective, or selected elective course in the program**
      Required course for BS-CS.

6. **Specific goals for the course**
   a. **Specific outcomes of instruction, ex. The student will be able to explain the significance of current research about a particular topic.**

<table>
<thead>
<tr>
<th>Course Learning Outcomes</th>
<th>Student Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ability to read and write medium sized programs</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability to understand syntax/semantics, identify and fix syntax errors, compile, build and execute medium sized programs</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Understand, design and implement abstract data types</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability to identify the need for, and declare and/or use classes, objects, inheritance, and polymorphism</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability to use input and output streams</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability for write code using a language API</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability to understand and use event driven programming for developing GUI</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability to develop and communicate its understanding and descriptions of systems, problems, and requirements using text and diagrams (UML) for documentation</td>
<td>1a, 1i</td>
</tr>
<tr>
<td>Ability to analyze and design UML class diagrams with many classes in an inheritance/composition hierarchy</td>
<td>1a, 1i, 2b</td>
</tr>
</tbody>
</table>
Ability to analyze GUI programs and draw the window hierarchies form a program  |  1a, 1b, 1i, 2b
---|---
Ability to identify the need for window listeners and layout managers  |  1a, 1i
Ability to design GUI interface using 2D layout and be able to convert to a program  |  1a, 1i
Understand and use existing classes to build bigger program components  |  1a, 1i
Ability to master basic software engineering concepts for object-oriented design and analysis  |  1a, 1b, 1i, 2b
Understand and use recursion  |  1a, 1i

b. Student Outcomes:
   - 1a: An ability to apply knowledge of computing and mathematics appropriate to the discipline
   - 1b: An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
   - 1i: An ability to use current techniques, skills, and tools necessary for computing practice.
   - 2b: An ability to apply design and development principles in the construction of software systems of varying complexity

7. Brief list of topics to be covered
   - Objects and Classes
   - Strings and Text I/O
   - Thinking in Objects
   - Inheritance and Polymorphism
   - GUI Basics
   - Exception Handling
   - Abstract Classes and Interface
   - Graphics
   - Event-driven Programming
   - Creating User Interface
   - Binary I/O
   - Recursion