1. **Course number and name**  
   **CSC340: Programming Methodology**

2. **Credits and contact hours**  
   3 credits  
   Contact hours: 150 minutes of lecture sessions /week

3. **Instructor’s or course coordinator’s name**  
   Course coordinator: Hui Yang, Associate Professor of Computer Science

4. **Text book, title, author, and year**  
   *C++ for Java Programmers (Paperback) by Mark Allen Weiss*, Prentice Hall

   **other supplemental materials**  
   Lecture Slides

5. **Specific course information**  
   - **brief description of the content of the course (catalog description)**
     
     This course explores advanced data structures and algorithms for manipulating them in C++. Emphasis is placed on design and implementation of those structures and a variety of practical applications. Algorithm coverage will include sorting and searching, and graph algorithms. Students will solve a series of problems to enhance their problem-solving skills.

   - **prerequisites or co-requisites**
     
     grades of C or better in CSC 220, CSC230 and Math 227. Concurrent enrollment in CSC 412 is recommended.

   - **indicate whether a required, elective, or selected elective course in the program**
     
     Required for Computer Engineering.

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.**

     At the end of this course students will
• Be able to write medium-sized C++ programs utilizing STL and an integrated development environment
• Determine which of the common sorting and searching algorithms to utilize for given problems
• Be able to apply and implement graph algorithms in practice

b. explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course.
   Course addresses ABET Student Outcome(s): a, b, c, e, j, k.

7. Brief list of topics to be covered
• C++ Topics – transitioning from Java
  STL, pointers, namespaces, inheritance, polymorphism, parameter passing, dynamic memory allocation
• Graph algorithms
  Searching and sorting algorithms
• Sorting: quick sort, bubble sort, binary sort, mergesort, heapsort and insertion sort; runtimes of these algorithms will be considered.