1. **Course Number and Name:**
   CSET 3150 Introduction to Algorithms
2. **Credits and Contact hours:**
   Credits: 4 hours, Contact: 3 lecture hours; 1 lab hour
3. **Instructor's or course coordinator's name:**
   Jared Oluoch
4. **Text book, title, author, and year:**
   Introduction the Algorithms, 3rd Edition, Thomas H. Gorman, 2009
   a. **Other supplemental materials:**
      - Various web references assigned by the instructor
      - "Data Structures and Algorithm Analysis in C++," 3rd Edition, Mark Alan Weiss, Addison-Wesley, ISBN 0-321-37531-9
      - "C++ Primer Plus," 5th Edition, Stephen Prata, Sams. November 2004. ISBN
5. **Specific Course Information:**
   a. **Brief description of the content of the course (catalog description):**
   This course covers object oriented programming and advanced algorithms. Topic includes C++ and OO concepts, algorithms and data structures as implemented in the C++ and Java programming languages. The final project is implemented in Java. This course is programming intensive and lays a firm foundation for student's OO programming skills.
   b. **Pre-requisites, or co-requisites:**
   EET 2230
6. **Specific goals for the course:**
   c. **Specific outcomes of instruction:**
   1. Be able to find an algorithm to solve the problem,
   2. Be able prove that the algorithm solves the problem correctly,
   3. Be able to prove that we cannot solve the problem any faster,
   4. Be able to implement the algorithm

   d. **Explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course: a, b, c, i, j, k**
   a. An ability to select and apply knowledge of computing and mathematics appropriate to the discipline. More specifically, an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
   b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
   c. An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs. More specifically, an ability to apply design and development principles in the construction of

software systems of varying complexity.

i. An ability to select and apply current techniques, skills, and tools necessary for computing practice.

j. An ability to conduct standard tests and measurements; to conduct, analyze, and interpret experiments; and to apply experimental results to improve processes.

k. A commitment to quality, timeliness, and continuous improvement.

7. **Brief list of topics to be covered:**
   1. Introduction
   2. Introduction to Sorting Algorithms
   3. Asymptotic notation
   4. Recurrences
   5. More on Sorting Algorithms (chapters 6-9)
   6. Searching Algorithms (chapters 11-13)
   7. Selection Algorithms
   8. Advanced Data Structures
   9. Dynamic Programming
   10. Greedy Algorithms
   11. Graph Algorithms (chapters 22-25)
   12. String matching
   13. NP-Complete Problems

1. **Course Number and Name:**
   CSET 4250 Applied Programming Languages
2. **Credits and Contact hours:**
   Credits: 3 hours, Contact: 3 lecture hours
3. **Instructor's or course coordinator's name:**
   Jared Oluoch
4. **Text book, title, author, and year:**
   Concepts of Programming Languages, 9th Edition, Robert W. Sebesta, 2009
   a. **Other supplemental materials:**
      None
5. **Specific Course Information:**
   a. **Brief description of the content of the course (catalog description):**
      This course teaches methodologies to select the most appropriate language for a specific engineering technology application. Topics include comparison of programming languages by evolution, formal specifications, structures, features, application domains, programming paradigms, implementation of syntax, semantics and program run-time behavior.
   b. **Pre-requisites, or co-requisites:**
      CSET 4100 and Junior Standing
6. **Specific goals for the course:**

a. **Specific outcomes of instruction:**
   1. Be able to explain and apply a broad range of concepts about programming languages.
   2. Be able to recognize, define, and make correct use of most common programming languages terminology.
   3. Design, implement, test, and debug simple programs in an object-oriented programming language, functional paradigm logical programming and scripting languages.
   4. Identify and describe the properties of a variable such as its associated address, value, scope, persistence, and size.

b. **Explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course: a, b, i, k**

   A. An ability to select and apply knowledge of computing and mathematics appropriate to the discipline. More specifically, an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.

   B. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.

   I. An ability to select and apply current techniques, skills, and tools necessary for computing practice.

   K. A commitment to quality, timeliness, and continuous improvement.

7. **Brief list of topics to be covered:**
   1. Introduction to Programming Languages
   2. Attribute Grammar and Static Semantics
   3. Describing Syntax and semantics
   4. Parsing
   5. Attributes of Variables, binding, scopes
   6. Data Types
   7. Perl Introduction
   8. Expressions Statements
   9. Statement-Level Control Structures
   10. Subprograms
   11. ADT and Encapsulation Constructs
   12. Object Oriented Programming
   13. Functional Programming Languages
   14. Logic programming and Prolog
   15. Concurrency, Exception
   16. Advanced topics: Programming Language Design