

CIS 255 WX FALL 2009 CSM PROGRAMMING METHODS I: JAVA

INSTRUCTOR Melissa Green greenm@smccd.edu <http://www.smccd.edu/accounts/greenm>
Office: 12-188F 650-574-6374 Office hours: MW 2:00-3:00 PM T 10:30-12:30 PM Th 3:30-4:30 PM

LECTURE CIS 255 WX CRN 88678 4 Units Thursday, 5:00-7:40 PM Room 18-304

LAB ONLINE

Plus One Hour by Arrangement

Students will use the textbook's CD-ROM, textbook web site, the Sun Java API and the World Wide Web to enrich their learning.

TEXT Java How to Program, 8th Edition by Deitel & Deitel Prentice Hall ISBN 0136053068

COURSE DESCRIPTION

Object-oriented programming methodology for both computer science majors and computer professionals. Systematic approach to design, construction, and management of computer programs; emphasizing program documentation, testing, debugging, maintenance, and software reuse. Also includes UML, virtual machines, exception handling, sorting and searching algorithms, recursion, fundamental graphics, and computer ethics. This course conforms to the ACM CS1 standards.

Prerequisites: Math120 (Intermediate Algebra) or equivalent, CIS 118/119 or CIS 254 (Introduction to Object-Oriented Program Design) or equivalent. Please note that these are serious prerequisites. CIS 255 is **NOT** intended as a first course in programming.

GRADING

Tests	45%	90-100	A	80-89	B
Final Exam	25%	70-79	C	60-69	D
Assignments	30%	0-59	F		

Students with a cumulative grade one point below cutoff will be promoted to the next highest grade if they have completed all assignments.

This course does allow "pass/no pass" grading. You must maintain a "C" average to pass.

There will be approximately 8 programming assignments as well as lab assignments. Programming assignments will be graded on program correctness, documentation, and style. There will be 4 online tests over the semester, each worth 75 points. Each test focuses on recent material but may also cover material from the beginning of the semester. The tests will be based on the textbook, handouts, and techniques you have used on related assignments. I will use your 3 highest test scores in determining your test grade. **There are NO makeup tests.**

OTHER IMPORTANT DATES

Tuesday, September 1, 2009	Last day to add or drop with eligibility for fee credit or refund
Sunday, September 6, 2009	Last day to complete WebSMART registration
Monday, September 7, 2009	Labor Day– NO CLASS
Friday, September 11, 2009	Last day to drop classes with no notation on student record
Wed., September 23, 2009	Last day to declare pass/no pass option
Thursday., November 12, 2009	Flex Day– NO CLASS
Wed., November 18, 2009	Last day to withdraw with a "W" on student record
November 26-29, 2009	Thanksgiving – NO CLASS

COMPLETING ASSIGNMENTS

This course will require **at least** twelve hours of computer work each week in addition to preparation time. All assignments must be uploaded to **WebAccess** by the due date/time. In addition, you must submit a printout of all source code for each assignment. Assignments will **NOT** be accepted by e-mail. Students are expected to do their own work. Any case of duplicate assignments will result in a grade of zero for all people involved. Late assignments will have a 50% penalty and are accepted only up to the beginning of the next class. You will receive a separate handout with programming guidelines.

PARTICIPATION

Under normal circumstances I do **NOT** drop students from the class rolls. It is the student's responsibility to file the paperwork needed to drop or withdraw from this class. If you simply stop participating in the class you will probably receive an "F".

FINAL EXAM SCHEDULE

The final exam covers all material for the semester.

CIS 255 WX Thursday, December 17, 2009 5:00-7:30 PM

STUDENT LEARNING OUTCOMES

Upon completion of this course, students should be able to

- Demonstrate understanding of the principal object-oriented programming concepts
- Employ Unified Modeling Language (UML) notation to model the object-oriented design of a non-trivial computer program
- Implement a medium-size computer program that is stylistically and functionally correct, based on an object-oriented design model.
- Reuse existing components through inheritance and polymorphism.
- Implement, test, and debug simple recursive methods.
- Explain and employ basic sorting and searching algorithms.
- Perform exception handling.
- Use and create standard API documentation for classes and methods.