Catalog Description:

**CMPS 134 - Computer Science I, 3 credits, (Co-requisite: CMPS 134L)**

An introduction to programming concepts and methodology using an object-oriented programming language (currently Java). Topics include problem analysis, abstraction, modularization, the development and use of algorithms, reuse, and the use of programming constructs including data types, classes, control structures, and methods.

**CMPS 134L - Computer Science I Lab, 1 credit, (Co-requisite: CMPS 134)**

Programming-related activities are undertaken that apply essential concepts from CMPS 134, including problem decomposition, modularization, flow of control, scoping, object-orientation, and algorithm development.

**Student Learning Outcomes:** Upon completion of the course, a successful student will have the ability to do each of the following:

- Explain the role of software in computer systems.
- Explain concepts and terminology of Object Oriented Software Development.
- Explain the purpose and use of an IDE (Integrated Development Environment) in the development of software.
- Read, understand and explain source program components written in the Java programming language.
- Readily make use of Java’s String class along with several other such classes from the Java Standard Library.
- Research and make use of other classes from the Java Standard Library, although they may not have been actually demonstrated or used during the course.
- Formulate an abstraction and express it as a complete Java class specification consisting of variables (instance and otherwise), and methods serving as constructors, observers, mutators, privates, and others as appropriate.
- Develop a Java application program to perform some well-defined task making appropriate use of user-defined classes as well as classes from the Java Standard Library.
- Make appropriate use of Java’s primitive (i.e. intrinsic) data types, operators and control structures, and furthermore have the ability to readily master those similarly available in other programming languages.
- Make appropriate use of the array data structure in the solution of problems and the specifications of tasks for a computer to carry out, and to appreciate that other data structures exist and may be applicable.
- Explain searching and sorting in general and be familiar with specific algorithms for each.
- Recognize the use of recursion and be able to explain it as an alternative to iteration.
- Recognize the use and potential value of inheritance in the design of software components.
- Recognize the use and potential value of interfaces in the design of software components.

REFERENCES: As presented and as required.

Course Web Site: http://www.cs.scranton.edu/~jackowit/public/Fall2018/c134
(This site serves as our primary electronic communication tool for this course. You will use it to access required and optional course material and to submit and review assignments. Initially, you must register to obtain full access to the functionality of this site.)

GRADING: Worth

Quizzes: (announced 1 class in advance) 20%
(Several (at least 5) short (10 min.) graded activities will be undertaken in class as a means of providing feedback to both instructor and student. It is planned that the lowest grade will be dropped in determining an average.)

Tests: (approximate date)
   Week of October 1st 20%

Final Exam: (combined; to be scheduled December 11th - 15th) 20%

Assignments:
   Programming, homework, etc. 40%
   Attendance and Class Participation considered.
(Your non-tardy attendance at all class meetings is expected, and so is your participation in these meetings. The accumulation of more than four absences may result in a diminished final course grade.)

PROCEDURES:

Lectures:
   • please sit in the same seat for every class meeting
   • feel free to ask and answer questions, and to contribute to discussions
   • classroom use of electronic devices/gadgets (including computers) is at the full discretion of the instructor. (Distracting others or yourself will not be tolerated.)

Tests and Quizzes:
   • always announced in advance, and no make-ups will be given
   • notice must be given if a test must be missed due to serious illness or emergency

Assignments:
   • assignments are activities accomplished outside of class meetings that require the development and submission of specified items (typically source programs) to be tested, evaluated and graded
   • each student is required to do his/her own work on each assignment
   • discussions and mutually beneficial collaboration among students is encouraged, but must not be to the point of representing someone else’s effort and understanding as your own as this would be considered to be academic dishonesty (See Academic Code of Honesty in the Student Handbook at http://catalog.scranton.edu/mime/media/view/42/5075/2018-2019-Student-HandbookFINAL.pdf)
   • academic dishonesty will be dealt with severely
   • each assignment will have a specified due date, and a specified deadline
   • normally the deadline is later than (typically, but not always two days) the due date
   • work submitted after the due date is considered to be “late”, will be accepted for grading but may be assessed a penalty (depending upon how late it is, and whether or not worthwhile preliminary work had been submitted by the due date).
   • work may not be submitted after the deadline; it is considered to be too late, may not be accepted for grading, and may receive a grade of zero (depending upon whether or not worthwhile preliminary work had been submitted prior to the due date or prior to the deadline).
   • incomplete work generally will receive a grade much higher than zero
   • work not submitted will receive a grade of zero

Other:
   • Obligations: http://www.scranton.edu/equity-diversity/Faculty-Resources.shtml