COCONINO COMMUNITY COLLEGE
COURSE OUTLINE

Created by: Jeff Jones October 30, 2013
Revised by: Jeff Jones January 26, 2017
Status: Permanent
Effective Term: Fall 2017

A. Identification:
   1. Subject Area: Computer Information Systems (CIS)
   2. Course Number: 215
   3. Course Title: Programming with Python
   4. Credit Hours: 3
   5. Course Description: Introduction to programming logic and structures as applied to business computer applications and programming languages through structured techniques and high-level languages. No prior computer experience necessary. General Education: Options. May be taken for S/U credit. Three lecture.

B. Course Goals: To provide students with a general introduction to computer programming and its place in society. Includes hands-on use of computers, python, and implementation of short programs and solutions.

C. Course Outcomes
   Upon successful completion of this course, student will:
   1. create flow-charts and pseudocode to solve real world problems;
   2. translate pseudocode and flow-charts into effective programming code;
   3. create internal and external program documentation;
   4. debug and maintain programs to meet changing requirements and challenges;
   5. generate functioning code using fundamental structured programming concepts;
   6. and describe the history of coding, ethical issues such as hacking, malware, security, and societal implications of computers and coding.

D. Course Outcomes Assessment will include:
   1. course grades determined by the instructor as outlined in the course syllabus
   2. pre and posttest assessing terminology and syntax;
   3. and critical thinking skills rubric applied to final coding project.

E. Course Content will include:
   1. structured programming concepts – abstract problem definition to code generation;
   2. programming syntax and documentation;
   3. object oriented programming vs. functional programming;
   4. logic structures: conditions, datatypes, lists, decisions, sequence, selection, repetition, arrays, etc;
   5. and structured design tools and standard processes such as abstraction, debugging, exception handling, etc.;
   6. proper documentation troubleshooting, performance, and security;
   7. and history of coding, ethical issues such as hacking, security, and societal implications of computers and coding.