Course Outline Report

CS 020 - Python Application Programming

COLLEGE:
Merritt College

ORIGINATOR: Brown, Courtney

DIVISION/DEPARTMENT:
Merritt - Division II/M - Technology

STATE CONTROL NUMBER: CCC000601581

DATES:
BOARD OF TRUSTEES APPROVAL DATE: 01/08/2019
STATE APPROVAL DATE: 01/31/2019
CURRICULUM COMMITTEE APPROVAL DATE: 11/29/2018
REQUISITE VALIDATION:

CURRENT EFFECTIVE DATE: 08/01/2019

1. REQUESTED CREDIT CLASSIFICATION:

D - Credit - Degree Applicable

N - Not Basic Skills

1 - Program Applicable

2. DEPT/COURSE NO:

CS 020

3. COURSE TITLE:

Python Application Programming

4. COURSE:

Course
MC New Course

TOP NO. 0707.10 - Computer Programming*

5. UNITS:

Variable No

Units (Min) 3.000

Min Total

Hours

Lecture Hours (Min) 2.000

35

Lab/Studio/Activity Hours (Min) 3.000

52.5

6. SELECTED TOPIC:

NO. OF TIMES OFFERED AS SELECTED TOPIC:

AVERAGE ENROLLMENT:

7. JUSTIFICATION FOR COURSE:

Python is an interpreted language that is distinguished by its large and active scientific computing community. This community creates computer tools to solve problems and distributes these tools as software libraries. Part of Python's success in scientific computing is the ease of integrating C, C++, Fortran, Java code through the use of software libraries. This is an introductory programming course intended to provide fundamental skills in creating and organizing maintainable Python programs. It enables people working primarily on non-computing domains (Natural Sciences, Linguistics, IT Operations) to use domain specific software libraries for exploratory computing, data analysis, and data visualization. Students will also gain exposure to tools commonly used to manage their development environment, share work products, and organize tools and libraries.

8. COURSE/CATALOG DESCRIPTION

Introduction to computer programming in Python 3: Control structures, algorithm design, and the integration of domain specific libraries (tensofflow, numpy, scipy) into a program; elements of good programming style and use of Object Oriented Programming (OOP) to manage complexity and Jupyter interactive notebooks to share results.

9. OTHER CATALOG INFORMATION

a. Modular: No

If yes, how many modules:

b. Open entry/open exit: No

c. Grading Policy: Both Letter Grade or Pass/No Pass

d. Eligible for credit by Exam: No

e. Repeatable according to state guidelines: No

f. Required for degree/certificate (specify):

g. Meets GE/Transfer requirements (specify): AA/AS area 4c

h. C-ID Number:

Expiration Date:

i. Are there prerequisites/corequisites/recommended preparation for this course? Yes

10. LIST STUDENT PERFORMANCE OBJECTIVES (EXIT SKILLS):

If an Objective cannot be deleted, make sure a Content-Review found in the Content Validation Page is not using that objective.

Objectives

1. **Design and implement computer programs using abstract data types, loops, classes, methods or functions.**
2. **Design and implement computer programs using objects, inheritance, polymorphism in addition to control structures.**
3. **Integrate software libraries into application programs.**
4. **Publish programs and summarize data as libraries or notebooks.**

11. COURSE CONTENT:

LECTURE CONTENT:
1. Introduction 10%
- Compiled Languages
- Interpreted Languages
- Configuring the program execution environment
- Version Dependencies, Version Pinning and System Stability
- The interactive computing environment

2. Control Structures for Programs 20%
- Console Input/Output
- Accumulators and Decision Making
- Arrays and Evaluating unsorted collections of iterms
- File input/Output of binary values
- Assigning Rank to Collections

4. Object Oriented Design 20%
- Design of programs as interacting objects
- Defining classes
- Using instances of objects
- Inheritance and overriding functions

5. Abstraction and Modeling 10%
- Abstract Data Types
- Implementing an algorithm
- Exception Handling
- Recursive Programming

6. Libraries for Data Structures and Data Sets 10%
- Pandas, Numpy, and Abstract Data Types
- Time Series Data
- Dataframes

- Gaps and Missing Data

7. Data Loading and Storage 15%
    - File Formats
    - Wrangling and Reshaping
    - Clean, Transform, Merge

8. Plotting and Visualization 5%
    - Aggregation
    - Group Operatioons

9. Financial and Economic Applications 10%
    - Cross-sectional data
    - Time series and cross-section alignment
    - Operating on Time Series with Different Frequencies
    - “Time of Day” and “As-of” data selection
    - Splicing data sources together

LAB CONTENT:
Consists of laboratory exercises to create programs which demonstrate the following programming abilities and techniques:

1. Compiling and executing a simple application program 10%

2. Using string and numeric variables 10%

3. Inputting data into variables and use in calculations 10%

4. visualizing and formatting output 10%

5. Management of interactive development environment and tool versions 10%

6. Data loading, transformation, decisions and storage 10%

7. Control structures for repetition and searching 10%

8. Handling unexpected data and condigtions 5%

9. Using modular program structure with subroutines and functions 10%

10. Creating arrays and Abstract Data Types 10%

11. Program debugging 5%

12. METHODS OF INSTRUCTION (List methods used to present course content):

- Activity
- Lecture
- Lab
- Observation and Demonstration
- Discussion

- Critique
- Projects
- Individualized Instruction
- Threaded Discussions

Other Methods:
Readings and exercises in textbook Programming solution demonstrations Lab and programming assignments Team programming projects

13. ASSIGNMENTS

Out-of-class Assignments (List all assignments, including library assignments. Requires two (2) hours of independent work outside of class for each unit/weekly lecture hour. Outside assignments are not required for lab-only courses, although they can be given.)

Override Outside Class Hours: No

Outside-of-Class Hours (Min) 4.000

Outside-of-Class Hours (Max) 0.000

Override Outside-of-Class Hours (Min) 0.000

Override Outside-of-Class Hours (Max) 0.000

Out of class Assignment
1. Assigned text readings.
2. Exercises from the textbook.
3. Programming assignments using Control Structures
   Console Input/Output
   Accumulators and Decision Making
   Arrays and Evaluating unsorted collections of iterms
   File input/Output of binary values
   Assigning Rank to Collections
4. Programming assignments using of Object Oriented Design
   Design of programs as interacting objects
   Using instances of objects
   Inheritance and overriding functions
   Implementing an algorithm
   Exception Handling
   Recursive Programming
5. Programming projects on using several different libraries
   Numpy
   Pandas
   Tensorflow

14. STUDENT ASSESSMENT: (Grades are based on):

- ESSAY (Includes "blue book" exams and any written assignment of sufficient length and complexity to require students to select and organize ideas, to explain and support the ideas, and to demonstrate critical thinking skills.)

- COMPUTATION SKILLS
- NON-COMPUTATIONAL PROBLEM SOLVING (Critical thinking should be demonstrated by solving unfamiliar problems via various strategies.)
- SKILL DEMONSTRATION
- MULTIPLE CHOICE
- OTHER (Describe)

OTHER (Describe):
Programming assignments/exercises, team programming projects.

15. TEXTS, READINGS, AND MATERIALS

A. Textbooks:
YesNo37
Gaddis, Tony. *Starting Out with Python*. 4 edition Pearson, 2015.
McKinney, Wes. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. 2 edition O'Reilly, 2017.

*Date is required: Transfer institutions require current publication date(s) within 5 years of outline addition/update.

B. Additional Resources:
Library/LRC Materials and Services:

The instructor, in consultation with a librarian, has reviewed the materials and services of the College Library/LRC in the subject areas related to the proposed new/updated course
Print Materials were reviewed? No

Non-Print Materials were reviewed? No

Online Materials were reviewed? No

Services were reviewed? No

Specific materials and/or services needed have been identified and discussed. Librarian comments:

C. Readings listed in A and B above are: (See definition of college level):
YesNo39
Primarily college level

16. DESIGNATE OCCUPATIONAL CODE:

C - Clearly Occupational

17. LEVEL BELOW TRANSFER:

Y - Not applicable

18. CALIFORNIA CLASSIFICATION CODE:

Y - Credit Course

19. NON CREDIT COURSE CATEGORY:

Y - Not Applicable, Credit course

20. FUNDING AGENCY CATEGORY:

Y - Not Applicable (funding not used to develop course)

REQUISITES AND ADVISORIES

RECOMMENDED PREPARATION:
CIS 005 Introduction to Computer Science and MATH 002 Precalculus with Analytic Geometry or MATH 013 Introduction to Statistics or MATH 203 Intermediate Algebra

STUDENT LEARNING OUTCOMES

1. **Select appropriate control and data programming structures, including variable definitions, alternation and iteration, functions and objects.**

   exam, essay, student project, written exercise, skill demonstration.

2. **Plan the design and implementation of a program and select appropriate programming algorithms for implementation.**

   exam, essay, student project, written exercise, skill demonstration

3. **Identify strategies to isolate programming defects formulate a plan to debug programming code.**

   exam, essay, student project, written exercise, oral presentation, skill demonstration