CNIT R161: PROGRAMMING ESSENTIALS IN PYTHON

Originator
alynch

College
Oxnard College

Discipline (CB01A)
CNIT - Computer Networking/IT

Course Number (CB01B)
R161

Course Title (CB02)
Programming Essentials in Python

Banner/Short Title
Programming Essentials Python

Credit Type
Credit

Start Term
Fall 2021

Catalog Course Description
How great would it be to write your own computer program or design a modern web or desktop application? Both are a possibility if you learn how to code in Python. Python is the very versatile, object-oriented programming language used by startups and tech giants, Google, Facebook, Dropbox and IBM. Python is also recommended for aspiring young developers who are interested in pursuing careers in security, networking, artificial intelligence (AI), machine learning, and Internet-of-Things. This course utilizes the Cisco Networking Academy PCAP Python curriculum.

Taxonomy of Programs (TOP) Code (CB03)
0708.10 - *Computer Networking

Course Credit Status (CB04)
D (Credit - Degree Applicable)

Course Transfer Status (CB05) (select one only)
A (Transferable to both UC and CSU)

Course Basic Skills Status (CB08)
N - The Course is Not a Basic Skills Course

SAM Priority Code (CB09)
C - Clearly Occupational

Course Cooperative Work Experience Education Status (CB10)
N - Is Not Part of a Cooperative Work Experience Education Program

Course Classification Status (CB11)
Y - Credit Course

Educational Assistance Class Instruction (Approved Special Class) (CB13)
N - The Course is Not an Approved Special Class

Course Prior to Transfer Level (CB21)
Y - Not Applicable

Course Noncredit Category (CB22)
Y - Credit Course

Funding Agency Category (CB23)
Y - Not Applicable (Funding Not Used)

Course Program Status (CB24)
1 - Program Applicable

General Education Status (CB25)
Y - Not Applicable

Support Course Status (CB26)
N - Course is not a support course

Field trips
May be required

Faculty notes on field trips; include possible destinations or other pertinent information
Possible destinations would be an IT shop, an IT managed service provider, or a public sector managed network.

Grading method
Letter Graded

Alternate grading methods
Credit by exam, license, etc.

Does this course require an instructional materials fee?
No

Repeatable for Credit
No

Is this course part of a family?
No

Units and Hours

Carnegie Unit Override
No

In-Class

Lecture
Minimum Contact/In-Class Lecture Hours
43.75
Maximum Contact/In-Class Lecture Hours
43.75

Activity

Laboratory
Minimum Contact/In-Class Laboratory Hours
26.25
Maximum Contact/In-Class Laboratory Hours
26.25

Total in-Class

Total in-Class
Total Minimum Contact/In-Class Hours
70
Total Maximum Contact/In-Class Hours
70

Outside-of-Class

Internship/Cooperative Work Experience

Paid

Unpaid

Total Outside-of-Class

Total Outside-of-Class
Minimum Outside-of-Class Hours
87.5
Maximum Outside-of-Class Hours
87.5

Total Student Learning

Total Student Learning
Total Minimum Student Learning Hours
157.5
Total Maximum Student Learning Hours
157.5

Minimum Units (CB07)
3
Maximum Units (CB06)
3

Student Learning Outcomes (CSLOs)

	Upon satisfactory completion of the course, students will be able to:
1	Summarize what distinguishes Python from other programming languages.
2	Create a computer program using the Python programming language.
3	Identify errors and problem-solve using an algorithmic approach.

Course Objectives

	Upon satisfactory completion of the course, students will be able to:
1	Summarize what distinguishes Python from other programming languages.
2	Identify basic programming concepts.
3	List what distinguishes the different versions of the Python programming language.
4	Use primitive data types and data structures offered by the development environment.
5	Implement standard modules provided by Python.
6	Use Boolean values to compare difference values and control the execution paths.
7	Identify the basic methods of formatting and outputting data offered by Python.
8	Apply core program control structures.
9	Write simple applications with the Python programming language that relate to a specific domain.

10	Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, and the definition of functions.
11	Choose an appropriate data structure for modeling a simple problem.
12	Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, and the definition of functions.
13	Test applications with sample data.

Course Content

Lecture/Course Content

1. Introduction
   a. Python popularity
   b. Versions of the Python programming language
   c. What is object oriented programming
   d. Creating a basic Python program
2. Modular Design
3. Control and Evaluation
   a. Basic concepts: interpreting and the interpreter, compilation and the compiler, language elements, lexis, syntax and semantics, Python keywords, instructions, indenting
   b. Literals: Boolean, integer, floating-point numbers, scientific notation, strings
   c. Operators: unary and binary, priorities and binding
   d. Numeric operators: ** * / % // + –
   e. Bitwise operators: ~ & ^ | << >>
   f. String operators: * +
   g. Boolean operators: not and or
   h. Relational operators (== != > >= <
   i. Assignments and shortcut operators
   j. Accuracy of floating-point numbers
   k. Basic input and output: input(), print(), int(), float(), str() functions
   l. Formatting print() output with end= and sep= arguments
   m. Conditional statements: if, if-else, if-elif, if-elif-else
   n. The pass instruction
   o. Simple lists: constructing vectors, indexing and slicing, the len() function
   p. Simple strings: constructing, assigning, indexing, slicing comparing, immutability
   q. Building loops: while, for, range(), in, iterating through sequences
   r. Expanding loops: while-else, for-else, nesting loops and conditional statements
   s. Controlling loop execution: break, continue
4. Data Aggregates
   a. Strings in detail: ASCII, UNICODE, UTF-8, immutability, escaping using the \ character, quotes and apostrophes inside strings, multiline strings, copying vs. cloning, advanced slicing, string vs. string, string vs. non-string, basic string methods (upper(), lower(), isxxx(), capitalize(), split(), join(), etc.) and functions (len(), chr(), ord()), escape characters
   b. Lists in detail: indexing, slicing, basic methods (append(), insert(), index()) and functions (len(), sorted(), etc.), del instruction, iterating lists with the for loop, initializing, in and not in operators,list comprehension, copying and cloning
   c. Lists in lists: matrices and cubes
   d. Tuples: indexing, slicing, building, immutability
   e. Tuples vs. lists: similarities and differences, lists inside tuples and tuples inside lists
   f. Dictionaries: building, indexing, adding and removing keys, iterating through dictionaries as well as their keys and values, checking key existence, keys(), items() and values() methods
5. Functions and Modules
   a. Defining and invoking your own functions and generators
   b. Return and yield keywords, returning results, the None keyword, recursion
   c. Parameters vs. arguments, positional keyword and mixed argument passing, default parameter values
   d. Converting generator objects into lists using the list() function
   e. Name scopes, name hiding (shadowing), the global keyword
   f. Lambda functions, defining and using
   g. Map(), filter(), reduce(), reversed(), sorted() functions and the sort() method
   h. The if operator

i. Import directives, qualifying entities with module names, initializing modules
j. Writing and using modules, the __name__ variable
k. Pyc file creation and usage
l. Constructing and distributing packages, packages vs. directories, the role of the __init__.py file
m. Hiding module entities
n. Python hashbangs, using multiline strings as module documentation

6. Classes, Objects, and Exceptions

a. Defining your own classes, superclasses, subclasses, inheritance, searching for missing class components, creating objects
b. Class attributes: class variables and instance variables, defining, adding and removing attributes, explicit constructorinvocation
c. Class methods: defining and using, the self parameter meaning and usage
d. Inheritance and overriding, finding class/object components
e. Single inheritance vs. multiple inheritance
f. Name mangling
g. Invoking methods, passing and using the self argument/parameter
h. The __init__ method
i. The role of the __str__ method
j. Introspection: __dict__, __name__, __module__, __bases__ properties, examining class/object structure
k. Writing and using constructors
l. Hasattr(), type(), issubclass(), isinstance(), super() functions
m. Using predefined exceptions and defining your own ones
n. The try-except-else-finally block, the raise statement, the except-as variant
o. Exceptions hierarchy, assigning more than one exception to one except branch
p. Adding your own exceptions to an existing hierarchy
q. Assertions
r. The anatomy of an exception object
s. Input/output basics: opening files with the open() function, stream objects, binary vs. text files, newline character translation, reading and writing files, bytearray objects
t. Read(), readinto(), readline(), write(), close() methods

7. Program Development Lifecycle
8. Requirements Determinants and Analysis

Laboratory or Activity Content

1. The fundamentals of computer programming, i.e. how the computer works, how the program is executed, how the programming language is defined and constructed, what the difference is between compilation and interpretation, what Python is, how it is positioned among other programming languages, and what distinguishes the different versions of Python.
2. The basic methods of formatting and outputting data offered by Python, together with the primary kinds of data and numerical operators, their mutual relations and bindings; the concept of variables and variable naming conventions; the assignment operator, the rules governing the building of expressions; the inputting and converting of data.
3. Boolean values to compare difference values and control the execution paths using the *if* and *if-else* instructions; the utilization of loops (*while* and *for*) and how to control their behavior using the *break* and *continue* instructions; the difference between logical and bitwise operations; the concept of lists and list processing, including the iteration provided by the *for* loop, and slicing; the idea of multi-dimensional arrays.
4. The defining and using of functions – their rationale, purpose, conventions, and traps; the concept of passing arguments in different ways and setting their default values, along with the mechanisms of returning the function's results; name scope issues; new data aggregates: tuples and dictionaries, and their role in data processing
5. Python modules: their rationale, function, how to import them in different ways, and present the content of some standard modules provided by Python; the way in which modules are coupled together to make packages; the concept of an exception and Python's implementation of exceptions, including the *try-except* instruction, with itsapplications, and the *raise* instruction; strings and their specific methods, together with their similarities and differences compared to lists.
6. The fundamentals of OOP (Object Oriented Programming) and the way they are adopted in Python, showing the difference between OOP and the classical, procedural approach; the standard objective features: inheritance, abstraction, encapsulation, and polymorphism, along with Python-specific issues like instance vs. class variables, and Python's implementation of inheritance; objective nature of exceptions; Python's generators (the *yield* instruction) and closures (the *lambda* keyword); the means Python developers can use to process (create, read, and write) files.

Methods of Evaluation

Which of these methods will students use to demonstrate proficiency in the subject matter of this course? (Check all that apply):

Problem solving exercises

Skills demonstrations
Written expression

Methods of Evaluation may include, but are not limited to, the following typical classroom assessment techniques/required assignments (check as many as are deemed appropriate):

Computational homework
Individual projects
Laboratory activities
Laboratory reports
Objective exams
Oral presentations
Problem-Solving Assignments
Quizzes
Reports/papers
Research papers
Skill tests

Instructional Methodology

Specify the methods of instruction that may be employed in this course

Audio-visual presentations
Computer-aided presentations
Collaborative group work
Class discussions
Distance Education
Demonstrations
Group discussions
Guest speakers
Instructor-guided use of technology
Internet research
Lecture

Describe specific examples of the methods the instructor will use:

1. Instructor will use the Cisco Academy provided PowerPoints to lecture on the Python programming language course topics.
2. The instructor will introduce labs and demonstrate lab solutions where appropriate.
3. The instructor may summarize current events as it relates to developments with the Python programming language.
4. Small group activities related to researching projects and application creation using Python.

Representative Course Assignments

Writing Assignments

1. Students are required to answer the questions in the lab assignments to demonstrate that they grasp the material.
2. Students are required to respond to questions posed at the current Oxnard College distance education portal. An example would be for a student to respond to a question explaining why Python is such a popular programming language for AI and machine learning.

Critical Thinking Assignments

1. Students will research when it is appropriate to borrow Python code and when it is appropriate to create custom code and then make a personal recommendation when presented with a real-world scenario.
2. Students will evaluate using Python for artificial intelligence purposes and share their personal concerns regarding machines gaining intelligence and making decisions on their own.

Reading Assignments

1. Research and read up on Python projects to determine a personal class Python project.
2. Utilize the web to read on current events and trends with the Python programming language.

Skills Demonstrations

1. Students will demonstrate their completed Python project to the class.

Other assignments (if applicable)

1. In order to prepare for the Python Institute PCAP certification, students will be required to answer certification preparation questions included in the Cisco Networking Academy Python course.

Outside Assignments

Representative Outside Assignments

1. Students are required to read and study the information in the assigned chapter of the Cisco Academy Python curriculum in between classes in order to be prepared for the lecture and classroom lab activities. A typical reading activity would be for the students to read the material on Python functions and then create and run Python functions.
2. Design a new web app using Python.
3. Create a PC based game using Python.

Articulation

C-ID Descriptor Number

ITIS 130

Status

Aligned

Comparable Courses within the VCCCD

CS M10P - Introduction to Computer Programming using Python Language

District General Education

A. Natural Sciences

B. Social and Behavioral Sciences

C. Humanities

D. Language and Rationality

E. Health and Physical Education/Kinesiology

F. Ethnic Studies/Gender Studies

CSU GE-Breadth

Area A: English Language Communication and Critical Thinking

Area B: Scientific Inquiry and Quantitative Reasoning

Area C: Arts and Humanities

Area D: Social Sciences

Area E: Lifelong Learning and Self-Development

CSU Graduation Requirement in U.S. History, Constitution and American Ideals:

IGETC

Area 1: English Communication

Area 2A: Mathematical Concepts & Quantitative Reasoning

Area 3: Arts and Humanities

Area 4: Social and Behavioral Sciences

Area 5: Physical and Biological Sciences

Area 6: Languages Other than English (LOTE)

Textbooks and Lab Manuals

Resource Type
Textbook

Description
Cisco Networking Academy. *Programming Essentials in Python*. Cisco Press in concert with the Python Institute, 2017

Resource Type
Software

Description
Edube interactive programming environment, Python Institute.

Distance Education Addendum

Definitions

Distance Education Modalities

Hybrid (51%–99% online)
Hybrid (1%–50% online)
100% online

Faculty Certifications

Faculty assigned to teach Hybrid or Fully Online sections of this course will receive training in how to satisfy the Federal and state regulations governing regular effective/substantive contact for distance education. The training will include common elements in the district-supported learning management system (LMS), online teaching methods, regular effective/substantive contact, and best practices.

Yes

Faculty assigned to teach Hybrid or Fully Online sections of this course will meet with the EAC Alternate Media Specialist to ensure that the course content meets the required Federal and state accessibility standards for access by students with disabilities. Common areas for discussion include accessibility of PDF files, images, captioning of videos, Power Point presentations, math and scientific notation, and ensuring the use of style mark-up in Word documents.

Yes

Regular Effective/Substantive Contact

Hybrid (1%–50% online) Modality:

Method of Instruction	Document typical activities or assignments for each method of instruction
Asynchronous Dialog (e.g., discussion board)	Topics will be presented for discussion with the opportunity to provide commentary and feedback on fellow student responses.
E-mail	Email will be used for individual interaction between professor and student, to send group email reminders of deadlines, to inform of upcoming course content.
Face to Face (by student request; cannot be required)	Face to face with students will take place at student request to discuss specific questions, issues, or concerns.
Video Conferencing	Zoom or comparable video conferencing software to lecture on course content, demonstrate lab assignments, answer student questions in real time, and provide student assistance on anything that is course related.
Other DE (e.g., recorded lectures)	Any real-time instruction will be recorded and available to students through the LMS.

Hybrid (51%–99% online) Modality:

Method of Instruction	Document typical activities or assignments for each method of instruction
Asynchronous Dialog (e.g., discussion board)	Topics will be presented for discussion with the opportunity to provide commentary and feedback on fellow student responses.
E-mail	Email will be used for individual interaction between professor and student, to send group email reminders of deadlines, to inform of upcoming course content.
Face to Face (by student request; cannot be required)	Face to face with students will take place at student request to discuss specific questions, issues, or concerns.
Video Conferencing	Zoom or comparable video conferencing software to lecture on course content, demonstrate lab assignments, answer student questions in real time, and provide student assistance on anything that is course related.
Other DE (e.g., recorded lectures)	Any real-time instruction will be recorded and available to students through the LMS.

100% online Modality:

Method of Instruction	Document typical activities or assignments for each method of instruction
Asynchronous Dialog (e.g., discussion board)	Topics will be presented for discussion with the opportunity to provide commentary and feedback on fellow student responses.
E-mail	Email will be used for individual interaction between professor and student, to send group email reminders of deadlines, to inform of upcoming course content.
Video Conferencing	Zoom or comparable video conferencing software to lecture on course content, demonstrate lab assignments, answer student questions in real time, and provide student assistance on anything that is course related.
Other DE (e.g., recorded lectures)	Any real-time instruction will be recorded and available to students through the LMS.

Examinations

Hybrid (1%–50% online) Modality

Online
On campus

Hybrid (51%–99% online) Modality

Online
On campus

Primary Minimum Qualification

COMPUTER INFORMATION SYS

Additional local certifications required

PCAP – Certified Associate in Python Programming. The course is preparing students for this certification so the instructor needs to hold this certification.

Review and Approval Dates

Department Chair

09/14/2020

Dean

09/15/2020

Technical Review

09/23/2020

Curriculum Committee

09/23/2020

Curriculum Committee

11/25/2020

CCCCO

MM/DD/YYYY

Control Number

CCC000599228

DOE/accreditation approval date

MM/DD/YYYY