Colorado State University GLOBAL CAMPUS

ITS320 BASIC PROGRAMMING

Credit Hours: 3
Contact Hours: This is a 3-credit course, offered in accelerated format. This means that 16 weeks of material is covered in 8 weeks. The exact number of hours per week that you can expect to spend on each course will vary based upon the weekly coursework, as well as your study style and preferences. You should plan to spend 14-20 hours per week in each course reading material, interacting on the discussion boards, writing papers, completing projects, and doing research.

Course Description and Outcomes

Course Description:

This course provides students with a detailed overview of fundamental programming, design, and testing concepts using Python. Students are introduced to the fundamentals of Python scripting and will become proficient in writing modular Python classes. At the core of class method development, students will write Python methods using lists, dictionaries, conditional logic, and looping controls.

Course Overview:

This course introduces students to the skills needed to plan and create their own Python applications, user interface design skills, object-oriented concepts, and planning tools such as pseudocode and flowcharts. Students also learn to create and manipulate variables, constants, strings, sequential access files, classes, dictionaries, lists, and arrays. Additionally, students will learn how to create programs using recursion, how to plot and visualize data, and how to develop object-oriented programs using inheritance. This course teaches programming concepts using a hands-on approach where students are able to gain mastery of these skills by performing practical exercises in each module and reviewing Lynda.com videos for visual instruction. After the successful completion of this course, students will be able to develop applications that are used in the workplace. The course, thus, prepares students for entry-level software programming jobs.

Course Learning Outcomes:

1. Configure Python 3 on a personal computer.
2. Explain terminology used in programming and the tasks performed by a programmer.
3. Develop applications using variables, constants, selection structure, and repetition structure.
4. Construct applications using function procedures, string manipulation, lists and dictionaries.
5. Compose basic object oriented programming in Python by working with classes.
6. Identify Python programming exceptions.
7. Develop modular Python programs that consist of multiple files.

Participation & Attendance

Prompt and consistent attendance in your online courses is essential for your success at CSU-Global Campus. Failure to verify your attendance within the first 7 days of this course may result in your withdrawal. If for some reason you would like to drop a course, please contact your advisor.

Online classes have deadlines, assignments, and participation requirements just like on-campus classes. Budget your time carefully and keep an open line of communication with your instructor. If you are having technical problems, problems with your assignments, or other problems that are impeding your progress, let your instructor know as soon as possible.

Course Materials

Textbook Information is located in the CSU-Global Booklist on the Student Portal.

Course Schedule

Due Dates

The Academic Week at CSU-Global begins on Monday and ends the following Sunday.

- **Discussion Boards:** The original post must be completed by Thursday at 11:59 p.m. MT and Peer Responses posted by Sunday 11:59 p.m. MT. Late posts may not be awarded points.
- **Opening Exercises:** Take the opening exercise before reading each week's content to see which areas you will need to focus on. You may take these exercises as many times as you need. The opening exercises will not affect your final grade.
- **Mastery Exercises:** Students may access and retake mastery exercises through the last day of class until they achieve the scores they desire.
- **Critical Thinking:** Assignments are due Sunday at 11:59 p.m. MT.

Weekly Reading and Assignment Details

Module 1

Readings

- Chapter 1 in *Programming in Python 3 with ZyLabs*
- Klein, B. (2017). *Execute a Python script*. Retrieved from https://www.python-course.eu/python3_execute_script.php
- Klein, B. (2017). *Input from the keyboard*. Retrieved from https://www.python-course.eu/python3_input.php
- Klein, B. (2017). *Structuring with indentation*. Retrieved from https://www.python-course.eu/python3_blocks.php

Opening Exercise (0 points)

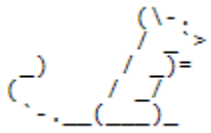
Mastery Exercise (10 points)

Discussion (25 points)

Critical Thinking (60 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Create a Python Application
Write a Python Program that outputs this mouse:



Assignment Instructions

1. Install Python3 on your computer if you do not have it already installed it. You can download it from https://www.python.org/downloads/release/python-362/.
2. Make sure you check the box to include the Python executable in your environment path.
3. Edit your Python program using your choice of editor such as Notepad, Notepad++, or Idle. Idle is a simple Python interactive development environment that installed with your Python package.
4. Save your Python code using the file name *ITS320_CTA1_ Option1.py.*
5. Execute your Python code in command prompt as *python ITS320_CTA1_ Option1*.

Assignment Deliverables

- Submit the text file named *ITS320_CTA1_Option1.py* that contains your Python code into the Module 1 drop box.

Option #2: Create a Python Application
Read two integers and print two lines. The first line should contain integer division, //, the second line should contain float division, /, and the third line should contain modulo division, %. You do not need to perform any rounding or formatting operations.

Input Format
The first line contains the first integer. The second line contains the second integer.

Output Format
Print the three lines as described above.

Sample Input
4
3

Sample Output
1
1.3333333333333333

Assignment Instructions

1. Install Python3 on your computer if you don't have it already installed. Make sure you check the box to include the Python executable in your environment path.
2. Edit your Python program using your choice of editor such as Notepad, Notepad++, or Idle. Idle is a simple Python interactive development environment that installed with your Python package.

3. Save your Python code in a file name *ITS320_CTA1_Option2.py*.
4. Execute your Python code in command prompt as *python ITS320_CTA1_Option2*.

Assignment Deliverables

- Submit the text file named *ITS320_CTA1_Option2.py* that contains your Python code into the Module 1 drop box.

Module 2

Readings

- Chapter 3 in *Programming in Python 3 with Zylabs*
- Klein, B. (2017). *Data types and variables*. Retrieved from https://www.python-course.eu/python3_variables.php

Opening Exercise (0 points)

Mastery Exercise (10 points)

Discussion (25 points)

Critical Thinking (65 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Creating a Python Application

Python has built-in string validation methods for basic data. It can check if a string is composed of alphabetical characters, alphanumeric characters, digits, etc.

```
str.isalnum()
This method checks if all the characters of a string are alphanumeric (a-z, A-Z and 0-9).
>>> print 'ab123'.isalnum()
True
>>> print 'ab123#'.isalnum()
False
str.isalpha()
This method checks if all the characters of a string are alphabetical (a-z and A-Z).
>>> print 'abcD'.isalpha()
True
>>> print 'abcd1'.isalpha()
False
str.isdigit()
This method checks if all the characters of a string are digits (0-9).
>>> print '1234'.isdigit()
True
>>> print '123edsd'.isdigit()
False
str.islower()
This method checks if all the characters of a string are lowercase characters (a-z).
>>> print 'abcd123#'.islower()
True
>>> print 'Abcd123#'.islower()
False
```

```
str.isupper()
This method checks if all the characters of a string are uppercase
characters (A-Z).
>>> print 'ABCD123#'.isupper()
True
>>> print 'Abcd123#'.isupper()
False
```

Assignment Instructions

Write a Python program that performs the following tasks:

1. Read from the console an arbitrary string S of length less than 50 characters.
2. In the first output line, print True if *S* has any *alphanumeric characters*. Otherwise, print False.
3. In the second line, print True if *S* has any *alphabetical characters*. Otherwise, print False.
4. In the third line, print True if *S* has any *digits*. Otherwise, print False.
5. In the fourth line, print True if *S* has any *lowercase characters*. Otherwise, print False.
6. In the fifth line, print True if *S* has any *uppercase characters*. Otherwise, print False.
7. Develop Python code that implements the program requirements.

Assignment Deliverables:

- Submit a text file containing your Python code into the Module 2 drop box. Name your file *ITS320_CTA2_Option1.py*.

Option #2: Creating a Python Application

Assignment Instructions

Develop a Python application that incorporates using appropriate data types and provides program output in a logical manner. Your program should prompt a user to enter a car brand, model, year, starting odometer reading, an ending odometer reading, and the estimated miles per gallon consumed by the vehicle. Store your data in a dictionary and print out the contents of the dictionary.

Assignment Submission Instructions:

- Submit a text file containing your Python code into the Module 2 drop box. Name your file *ITS320_CTA2_Option2.py*.

Module 3

Readings

- Chapter 4 in *Programming in Python with Zylabs*
- Klein, B. (2017). *Conditional statements*. Retrieved from https://www.python-course.eu/python3_conditional_statements.php

Opening Exercise (0 points)

Mastery Exercise (10 points)

Discussion (25 points)

Critical Thinking (65 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Creating a Program to Calculate the Value of a Ferrari

Assignment Instructions

Implement a program that reads in a year and outputs the approximate value of a Ferrari 250 GTO in that year. Use the following table that describes the estimated value of a GTO at different times since 1962.

Year	Value
1962-1964	$18,500
1965-1968	$6,000
1969-1971	$12,000
1972-1975	$48,000
1976-1980	$200,000
1981-1985	$650,000
1986-2012	$35,000,000
2013-2014	$52,000,000

(Source: *Programming in Python 3 with Zylabs, Chapter 4, Participation Activity 4.3.5)*

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 3 drop box. Name your file *ITS320_CTA3_Option1.py*.

Option #2: Creating a Program to Calculate Weekly Average Tax Withholding

Assignment Instructions

1. Create a program that will calculate the weekly average tax withholding for a customer, given the following weekly income guidelines:
   - Income less than $500: tax rate 10%
   - Incomes greater than/equal to $500 and less than $1500: tax rate 15%
   - Incomes greater than/equal to $1500 and less than $2500: tax rate 20%
   - Incomes greater than/equal to $2500: tax rate 30%

2. Store the income brackets and rates in a dictionary.
3. Write a loop that prompts the user for an income, looks up the tax rate from the dictionary, and prints the income, tax rate, and tax. The loop runs indefinitely until the user enters a loop termination sentinel.
4. Develop Python code that implements the program requirements.

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 3 drop box. Name your file *ITS320_CTA3_Option2.py*.

Module 4

Readings

- Chapter 5 in *Programming in Python 3 with ZyLabs*
- Klein, B. (2017). *Python loops*. Retrieved from https://www.python-course.eu/python3_loops.php

Opening Exercise (0 points)

Mastery Exercise (10 points)

Discussion (25 points)

Critical Thinking (60 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Repetition Control Structure – Five Floating Point Numbers

Assignment Instructions

Write a program that utilizes a loop to read a set of five floating-point values from user input. Ask the user to enter the values, then print the following data:

- Total
- Average
- Maximum
- Minimum
- Interest at 20% for each original value entered by the user.
    - Use the formula: Interest_Value = Original_value + Original_value*0.2

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 4 drop box. Name your file *ITS320_CTA4_Option1.py*.

Option #2: Repetition Control Structure – Grade Statistics

Assignment Instructions

1. Write a program that will provide important statistics for the grades in a class. The program will utilize a loop to read five floating-point grades from user input.
2. Ask the user to enter the values, then print the following data:
    - Average
    - Maximum
    - Minimum

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 4 drop box. Name your file *ITS320_CTA4_Option2.py*.

Module 5

Readings

- Chapters 6 & 7 in *Programming in Python 3 with ZyLabs*

Opening Exercise (0 points)

Mastery Exercise (10 points)

Discussion (25 points)

Critical Thinking (60 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: String Values in Reverse Order

Assignment Instructions

1. Write a Python function that will accept as input three string values from a user. The method will return to the user a concatenation of the string values in reverse order. The function is to be called from the main method.
2. In the main method, prompt the user for the three strings.

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 5 drop box. Name your file *ITS320_CTA5.Option1.py*.

Option #2: Third String in Reverse Order

Assignment Instructions

1. Write a Python function that will accept as input three string values from a user. The method will return to the user a concatenation of the first two strings and will print the third string in reverse order. The function is to be called from the main method.
2. In the main method, prompt the user for the three strings.

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 5 drop box. Name your file *ITS320_CTA5.Option2.py*.

Portfolio Milestone (25 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Corrections to Source Code Listings for Modules 1, 2, and 3

Assignment Instructions

1. Use the feedback from your instructor on your coding exercises from Modules 1, 2, and 3 to make corrections to your code as needed.
2. Assemble complete source code listings of all programs including any corrections you had to make for each correctly functioning program.

Assignment Submission Instructions

- Submit your text file(s) containing your Python code corrections for Modules 1, 2, and 3 in the Module 5 drop box.

Option #2: Corrections to Source Code Listings for Modules 1, 2, and 3)

Assignment Instructions

1. Use the feedback from your instructor on your coding exercises from Modules 2 and 3 to make corrections to your code as needed.
2. Assemble complete source code listings of all programs including any corrections you had to make for each correctly functioning program.

Assignment Submission Instructions

- Submit your text file(s) containing your Python code corrections for Modules 1, 2, and 3 in the Module 5 drop box.

Module 6

Readings

· Chapters 8 & 9 in *Programming in Python 3 with ZyLabs*

Opening Exercise (0 points)

Mastery Exercise (10 points)

Discussion (25 points)

Critical Thinking (60 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Working with Python Classes

For this assignment, you are given two complex numbers. You will print the result of their addition, subtraction, multiplication, division, and modulus operations. The real and imaginary precision part should be correct up to two decimal places.

Input Format

One line of input: The real and imaginary part of a number separated by a space.

Output Format

For two complex numbers and the output should be in the following sequence on separate lines:

- C + D
- C – D
- C * D
- C / D
- mod(C)
- mod(D)

For complex numbers with non-zero real and complex part, the output should be in the following format:

```
A + Bi
```

Replace the plus symbol (+) with a minus symbol (-) when B < 0.
For complex numbers with a zero complex part, i.e. real numbers, the output should be:

```
A + 0.00i
```

For complex numbers where the real part is zero and the complex part is non-zero, the output should be:

```
0.00 + Bi
```

Sample Input

```
2 1
```

```
5 6
```

Sample Output

```
7.00+7.00i
-3.00-5.00i
4.00+17.00i
0.26-0.11i
2.24+0.00i
7.81+0.00i
```

Assignment Instructions

1. Create a *Complex* class to perform the processing. Use the following code template.

```
import math

class Complex(object):
    def __init__(self, real, imaginary):
        self.real = real
        self.imaginary = imaginary

    def __add__(self, no):
                # enter your code here
        return Complex(real, imaginary)

    def __sub__(self, no):
        # enter your code here
        return Complex(real, imaginary)

    def __mul__(self, no):
        # enter your code here
        return Complex(real, imaginary)

    def __div__(self, no):
        # enter your code here
        return Complex(real, imaginary)

    def mod(self):
        # enter your code here
        return Complex(real, 0)

    def __str__(self):
        # enter your code here
        return result

# put this code in a main method
C = map(float, raw_input().split())
D = map(float, raw_input().split())
x = Complex(*C)
y = Complex(*D)
print '\n'.join(map(str, [x+y, x-y, x*y, x/y, x.mod(), y.mod()]))
```

2. Develop Python code that implements the program requirements.

Assignment Submission Instructions

- Submit the text files containing your Python code as a zip file into the drop box. Name your file *ITS320_CTA6.Option1.zip*.

Option #2: List Computations

Assignment Instructions

Compute their Cartesian product, *AxB* of two lists. Each list has no more than 10 numbers.

For example, given the two input lists:
A = [1,2]
B = [3,4]

then the Cartesian product output should be:
AxB = [(1,3),(1,4),(2,3),(2,4)]

Assignment Submission Instructions

- Submit a text file containing your Python code into the Module 4 drop box. Name your file *ITS320_CTA6_Option2.py*.

Module 7

Readings

- Chapter 10 in *Programming in Python 3 with ZyLabs*

Opening Exercise (0 points)

Discussion (25 points)

Mastery Exercise (10 points)

Portfolio Milestone (25 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Corrections to Source Code Listings for Modules 4, 5, and 6

Assignment Instructions

1. Use the feedback from your instructor on your coding exercises from Modules 4, 5, and 6 to make corrections to your code as needed.
2. Assemble complete source code listings of all programs including any corrections you had to make for each correctly functioning program.

Assignment Submission Instructions

- Submit your text file(s) containing your Python code corrections for Modules 4, 5, and 6 in the Module 7 drop box.

Option #2: Corrections to Source Code Listings for Modules 4, 5, and 6

Assignment Instructions

1. Use the feedback from your instructor on your coding exercises from Modules 4, 5, and 6 to make corrections to your code as needed.
2. Assemble complete source code listings of all programs including any corrections you had to make for each correctly functioning program.

Assignment Submission Instructions

- Submit your text file(s) containing your Python code corrections for Modules 4, 5, and 6 in the Module 7 drop box.

Module 8

Readings

- Chapter 11 in *Programming in Python 3 with ZyLabs*
- Klein, B. (2017). *Modular programming and modules*. Retrieved from https://www.python-course.eu/python3_modules_and_modular_programming.php

Opening Exercise (0 points)

Mastery Exercise (25 points)

Discussion (25 points)

Portfolio Project (300 points)

Choose one of the following two assignments to complete this week. Do not do both assignments. Identify your assignment choice in the title of your submission.

Option #1: Program Corrections, Lessons Learned, and Vehicle Inventory Program

Your portfolio project consists of three components:

1. Program corrections
2. Lessons learned reflection
3. Final program

Assignment Instructions

Program corrections:

1. Make appropriate corrections to all the programming assignments submitted as Critical Thinking Assignments from Modules 1-6.
2. Submit the programs along with a carefully outlined description of corrections made in order for programs to run correctly.

Lessons learned reflection:

- Create a 2-3 page summary that outlines the lessons learned in this Basic Programming course.

Final program:

Create a final program that meets the requirements outlined below.

1. Create an automobile class that will be used by a dealership as a vehicle inventory program. The following attributes should be present in your automobile class:
    - private string make
    - private string model
    - private string color
    - private int year
    - private int mileage

    Your program should have appropriate methods such as:
    - constructor
    - add a new vehicle
    - remove a vehicle

- update vehicle attributes

At the end of your program, it should allow the user to output all vehicle inventory to a text file.

2. Your final program submission materials must include your source code and screenshots of the application executing the application and the results.

Assignment Submission Instructions

Zip up the following files and submit in one file:

- Compiled Module 1-6 programs with corrections
- Lessons learned reflection
- Final program course code and application screenshots

Option #2: Program Corrections, Lessons Learned, and Home Inventory Program

Your portfolio project consists of three components:

1. Program corrections
2. Lessons learned reflection
3. Final program

Assignment Instructions

Program corrections:

1. Make appropriate corrections to all the programming assignments submitted as Critical Thinking Assignments from Modules 1-6.
2. Submit the programs along with a carefully outlined description of corrections made in order for programs to run correctly.

Lessons learned reflection:

- Create a two- or three-page summary that outlines the lessons learned in this Basic Programming course.

Final program:

1. Create a home inventory class that will be used by a National Builder to maintain inventory of available houses in the country. The following attributes should be present in your home class:
    - private int squarefeet
    - private string address
    - private string city
    - private string state
    - private int zipcode
    - private string Modelname
    - private string salestatus (sold, available, under contract)

    Your program should have appropriate methods such as:
    - constructor
    - add a new home
    - remove a home
    - update home attributes

2. At the end of your program, be sure that it allows the user to output all home inventory to a text file.
3. Be sure that your final program includes your source code and screenshots of the application, executing the application, and the results.

Assignment Submission Instructions

Zip up the following files and submit in one file:

- Compiled Module 1-6 programs with corrections
- Lessons learned reflection
- Final program course code and application screenshots

COURSE POLICIES

Grading Scale	
A	95.0 – 100
A-	90.0 – 94.9
B+	86.7 – 89.9
B	83.3 – 86.6
B-	80.0 – 83.2
C+	75.0 – 79.9
C	70.0 – 74.9
D	60.0 – 69.9
F	59.9 or below

Course Grading

20% Discussion Participation
37% Critical Thinking Assignments
35% Final Portfolio Project
8% Mastery Exercises

IN-CLASSROOM POLICIES

For information on late work and incomplete grade policies, please refer to our **In-Classroom Student Policies and Guidelines** or the Academic Catalog for comprehensive documentation of CSU-Global institutional policies.

Academic Integrity
Students must assume responsibility for maintaining honesty in all work submitted for credit and in any other work designated by the instructor of the course. Academic dishonesty includes cheating, fabrication, facilitating academic dishonesty, plagiarism, reusing /re-purposing your own work (see *CSU-Global Guide to Writing and APA Requirements* for percentage of repurposed work that can be used in an assignment), unauthorized possession of academic materials, and unauthorized collaboration. The CSU-Global Library provides information on how students can avoid plagiarism by understanding what it is and how to use the Library and Internet resources.

Citing Sources with APA Style
All students are expected to follow the *CSU-Global Guide to Writing and APA Requirements* when citing in APA (based on the APA Style Manual, 6th edition) for all assignments. For details on CSU-Global APA style, please review the APA resources within the CSU-Global Library under the "APA Guide & Resources" link. A link to this document should also be provided within most assignment descriptions in your course.

Disability Services Statement
CSU–Global is committed to providing reasonable accommodations for all persons with disabilities. Any student with a documented disability requesting academic accommodations should contact the Disability Resource Coordinator at 720-279-0650 and/or email ada@CSUGlobal.edu for additional information to coordinate reasonable accommodations for students with documented disabilities.

Netiquette
Respect the diversity of opinions among the instructor and classmates and engage with them in a courteous, respectful, and professional manner. All posts and classroom communication must be conducted in accordance with the student code of conduct. Think before you push the Send button. Did you say just what you meant? How will the person on the other end read the words?

Maintain an environment free of harassment, stalking, threats, abuse, insults or humiliation toward the instructor and classmates. This includes, but is not limited to, demeaning written or oral comments of an ethnic, religious, age, disability, sexist (or sexual orientation), or racist nature; and the unwanted sexual advances or intimidations by email, or on discussion boards and other postings within or connected to the online classroom. If you have concerns about something that has been said, please let your instructor know.