Course Syllabus Programming I
Fall 2017

Course Description

This course will emphasize *systems thinking* as an approach to solving computer problems and understanding formal logic. Programming theory and logic will be presented with hands-on practice in model environments, while students are provided with essential problem-solving methods, techniques and disciplines. Control flow, data manipulation and planning methods will be emphasized. Students will develop confidence in applying programming solutions, will be exposed to pertinent terminology, and will learn the effective use of reference materials. In this course we will start by learning to program in the Python programming language. Using the Python environment you will develop programming skills such as all of the basic programming constructs common to most modern programming languages specifically, sequential execution, program decision making, and code repetition. We will also use Python as an introduction to the object oriented programming concepts of classes, methods, inheritance and polymorphism. Once you have a firm handle on Python, we will use it to implement a variety of algorithms to solve problems in the area of Mathematics, Science, and Computer Simulation. Finally we will learn about the field of Game Programming including graphics, sound, algorithm development and game engines.

Course Requirements

- All program source code will be documented
- All program source code will contain comments listing the author, assignment number/title, and date
- You are required to keep a notebook for this course. Your notebook shall contain the following:
    - All class handouts
    - All lab exercises and source code
    - All quizzes
    - All tests
    - All homework assignments
- Your notebook will have a section for each category of item as described above. All material placed in your notebook will be dated and in order
- At sometime during the term I will review your notebook with you and you will receive a notebook grade.
- All program source code will be kept in a notebook in order.

Desired Learning Outcomes

- Students will be able to identify how a user interacts with a system
- Students will be able to utilize various problem solving techniques
- Students will be able to distinguish between different methods of mapping a program
- Students will be able to implement different methods of program flow
- Students will be able to effectively use different data types and variables

Class Methodology

The class will be comprised of a combination of lecture, discussion, exercise, reading, and projects. Students are expected to come to class each day fully prepared to participate in the day's activities. We will start each class at your desk where I will review with you the day's activities. In this course, you will be apply fundamentals that you learn by developing solutions to a variety of programming challenges.

Textbook

Various handouts and e-Books

Dual Enrollment Credit

Credit for CIS112 – Introduction to Object Oriented Programming is offered for this course through Great Bay Community College.

This course will provide a gentle, yet intense, introduction to programming using Python for highly motivated students with little or no prior experience in programming. The course will focus on planning and organizing programs, as well as the grammar of the Python programming language.

Topic List

1. Introduction
   a. Computer programs
   b. What is a computer
   c. The Python Programming Language
   d. The Python Programming Environment
   e. Flowcharts and Pseudocode
   f. Program Errors
   g. Algorithm Design
2. Programming with Numbers and Strings
   a. Variables
   b. Arithmetic
   c. Problem Solving: Hand example
   d. Strings
   e. Input and Output
   f. Graphics
3. Decision Making
   a. The if statement
   b. Relational operators
   c. Nested branches
   d. Problem solving: flowcharts
   e. Problem solving: test cases
   f. Boolean variables and operators
   g. Analyzing strings
   h. Input validation
4. Loops
   a. The while loop
   b. Problem solving: hand tracing
   c. Processing sentinel values
   d. Problem solving: storyboards
   e. Common loop algorithms
   f. The for loop
   g. Nested loops
   h. Processing strings
   i. Random numbers and simulation

5. Functions
    a. Black box approach
    b. Implementing and testing functions
    c. Parameter passing
    d. Return values
    e. Functions without return values
    f. Problem Solving: Reusable functions
    g. Problem Solving: Stepwise refinement
    h. Variable scope
    i. Recursive functions
6. Lists
    a. Basic properties of lists
    b. List operations
    c. Common list algorithms
    d. Using lists with functions
    e. Problem Solving: Adapting algorithms
    f. Tables
7. Files
    a. Reading and Writing text files
    b. Text input and output
8. Sets and Dictionaries
    a. Sets
    b. Dictionaries
9. Objects and Classes
    a. Object Oriented Programming
    b. Implementing a simple class
    c. Specifying a class interface
    d. Data representation
    e. Constructors
    f. Implementing class methods
    g. Testing a class
    h. Object references
10. Inheritance
    a. Inheritance hierarchies
    b. Implementing subclasses
    c. Calling the superclass constructor
    d. Overriding methods
    e. Polymorphism

Course Grading

Participation	3 Daily Project Grades
Notebook	4 Daily Project Grades
Homework	1 Daily Project Grade
Quizzes	1 Daily Project Grade
Projects/Labs	Daily Project Grade

Grading Standards

It is expected that you do your best on all project/lab activities. I generally assign one of four possible grades to your labs.

- √+ all project requirements completed in an exemplary manner (100)
- √ most project requirements completed in an acceptable manner (85)
- √- minimal project requirements completed in a substandard manner (70)
- 0 project not completed or completed in an unacceptable manner (0)

Assignment Due Dates

All assignments are due on the day they are assigned unless otherwise stated. Late assignments will be penalized one grade for each two days they are late. No assignments will be accepted after 5 school days late.

Tentative Schedule

Week	Topic
1	Getting Started with Python Variables and Memory
2	Basic Math Data Types Input and Output in Python
3	Basic Graphics
4	Decision Structures
5	Repetition Structures
6	Lists
7	Graphical User Interfaces - Using TKinter
8	Functions
9	Functions
10	Files
11	Sets and Dictionaries
12	Objects and Classes
13	Objects and Classes
14	Inheritance
15	Graphics - using PyGame Collision Detection Events Sound in Python
16	Visual Python
17	Scientific Programming

Effective: July 4, 2017

18	Mathematical Programming
19	Simulation
20	Game Programming

Classroom Rules

No food at computers. If you need a snack, eat it at the desk.
Start each session at the desk (not at computers)
Drinks are allowed but they must be in a container that is capable of being capped
No Internet use except for specific class assignments
No installation of programs; this includes the installation of files from USB drives.
No games!!!
Keep a class notebook of notes, handouts, assignments, tests, quizzes, etc.
Bring a writing instrument each day to class.
If you must leave the classroom, you must sign out.
If you wish to go to the school store, you must have a store pass.
Only 3 people out of the classroom at any time.
If you make a mess, clean it up.
See me for any exceptions to the above rules

Things you need to buy for class:

- One 1-inch notebook (for Professional Portfolio)
- One 3-inch notebook for notes and handouts
- Flash Drive for transporting files

Plagiarism

When working on various projects in this class, you are encouraged to collaborate at some level. Plagiarism is offering someone else's work as your own, whether one sentence, whole paragraphs, or blocks of code. Whether from an internet source, book, periodical, the writing of other students, or source code shared over the network. It is also dishonest to submit your own paper (or program) as original work in more than one course. There is a fine line sometimes between plagiarism and collaboration. Plagiarism is unacceptable here or at any time in your future career. Plagiarism will not be tolerated. In this class, plagiarized work will receive a grade of 0. **It is unacceptable for one person to write a program and share it over the network with other students placing your name on code that you did not write.** If you have any question as to whether or not you are plagiarizing someone's work, ask!!! Better ask for clarification than receive a 0 on your test.

Things you need to do:

- Install Python 3 on your home computer