Syllabus for MATH 175-01

Computer Programming for the Sciences, Spring 2016

Professor: Dr. James Rohal

Office:	Main Hall 219
Phone:	804-557-0425
Email:	james.rohal@westliberty.edu
Office Hours:	See `http://jamesrohal.com/schedule/`

Description: An introduction to programming using a modern programming language. Topics include primitive data types; stream and file I/O; Boolean expressions; control structures; functions; function overloading; recursion, multidimensional arrays; strings; and an introduction to dynamic memory management. Emphasis on procedural-oriented programming.

Prerequisite: Content knowledge of Precalculus.

Course Objectives:

- Understand fundamentals of programming such as instantiating variables, conditional and iterative execution, methods, etc.
- Understand fundamentals of object-oriented programming in Python, including defining classes, invoking methods, using class libraries, etc.
- Be aware of the important topics and principles of software development.
- Be able to use Python to create, debug, and run programs.
- Develop the ability to write a computer program to solve applied problems.
- Understand the limitations of procedural-oriented programming.

Required Materials:

(1) The textbook used is an online textbook called a zyBook through Zyante and costs $48. Go to `https://zybooks.zyante.com/#/students`, create an account, and use the zyBook code WestLibertyMATH175Spring2016.

(2) The main programming environment will be Python 3.x. This software is free and available from `http://python.org`. The software will be available on the computers in Main Hall 206. If you choose to install this on your own computer, make sure that you get version 3.x of Python, not version 2.x! The current version of Python is 3.5.1. Apple Macintosh computer owners must be especially careful because your computer may come with version 2.x preinstalled. You still need to install Python 3.x and may need to set up your environment differently. Come see me for the details.

(3) You need to choose an integrated development environment (IDE) for Python. We will be using the IDE provided with the Python installer called IDLE. Another recommended IDE is called is WingIDE 101: `https://wingware.com/downloads/wingide-101`. There are many other alternatives.

Attendance: Attendance is mandatory and is part of your grade. I will pass a seating chart around the first week of class. You are expected to sit in your assigned seat the remainder of the semester and attend all classes on time. Arriving late for a class or leaving early is very disruptive of class. If you need to leave early, please let me know at the beginning of class.

Homework: Homework problems are built in to each section of your zyBook. These are either called Participation Activities or Challenge Activities and help with the understanding of the material covered in each section. You are expected to complete approximately 90% of these activities. The due dates for the homework will be announced on Sakai and are found under the Assignment tab.

Assignments & Labs:

- A lab is a larger homework assignment that will cover multiple sections. Throughout the semester we will take class time to work on these labs. The lab and any relevant files are to be submitted electronically to me via email at james.rohal@westliberty.edu. The instructions and due dates for the labs will be announced on Sakai and are found under the Assignment tab.

- Programming assignments are large programming projects that incorporate multiple concepts. These programming assignments will function as the exams for this class. The assignment and any relevant files are to be submitted electronically to me via email at james.rohal@westliberty.edu. These assignments will be announced on Sakai and are found under the Assignment tab.

- Please do all assignments exactly as they are assigned. Do not change the problem to something you like better. Part of the assignment is to see if you can meet the specifications as given. If you want to use something that has not been discussed in class, *ask me first.* If you do use more advanced statements without permission, you will be penalized up to 75% on the program grade. This is not meant to stifle your creativity. The mantra of, "You have to learn the rules to break them properly," applies here. A good programmer uses simplicity where it works - there are enough bugs without introducing unnecessary complications.

- Turn in complete assignments and programs. If you don't have an assignment finished, wait until you are ready to turn in everything you are going to do. If a partial project is turned in, it will be graded as it stands. If more of the project is turned in later on, the entire project is considered as late as the date of the last part turned in, and the entire project is subject to be regraded and to late penalties.

- *Program Designs*: A design or an algorithm is a rough draft or outline of a program. It is important to do a design before you actually write the program. You will be expected to write a design (in pseudocode) for all programming assignments to go along with your implementation. Your design grade will be part of the programming assignment grade.

- *Program Test Plans*: A test plan is a set of test cases that you have come up with to test your program against. Each case has a description of what it is testing, inputs that cause that case to happen, and the expected output/outcome of that input. You devise these to make sure that all parts of your program are executed and tested in different situations. You should use your test plan later when your implementation is done.

- *Program Documentation*: Documentation is important to any program. With good documentation, you can go back six months later, pick up the program and understand what you did and why you did it. Comments should not just repeat code; they should explain why the code is the way it is. A programming standard which you are expected to follow will be described and developed during the course. It is posted on Sakai under Resources/Handouts. Programs can lose points for poor documentation or style, even if the program works.

Revise/Resubmit Cycle: To ensure that everyone masters the programming assignments, we will use an iterative feedback process. If one of the objectives below is not properly met, I will give you feedback and expect you to revise and resubmit. I will allow two resubmissions per assignment. In grading your code, we will focus on the following issues in order:

- Program design
- Correct function specifications and/or formatting
- Adequate test cases
- Correctness of the code (does it pass our test cases?)

Formatting is graded according to the course style guidelines, available under Resources/Handouts on Sakai. If your assignment fails one of the four tests above, I will notify you and ask you to resubmit. I stop checking once I find the first few errors, so you should not assume that the errors I point out are the only errors present.

Make-up Policy: If you do not turn an assignment the day it is due, then the score for that item will be zero unless you and I discuss it, and we both agree that a make-up is appropriate. Adjustments will be made for students who must miss the date due to illness, observance of a religious holiday, and for students who must due to a university sponsored activity (with letter from coach, sponsor, etc). I am more willing to give make-ups if *prior* permission is obtained.

Grading:

Homework	80 points
Labs	12 × 35 points
Programming Assignments	4 × 90 points
Final Programming Assignment	90 points
Attendance	$\begin{cases} 50 \text{ points}, & \text{if \# of abscences} = 0, 1, \text{ or } 2 \\ 50 - 10(\text{\# of absences} - 2) \text{ points}, & \text{if \# of absences} > 2 \end{cases}$

The standard grading scale will be used.

Special Attention: If you have a disability that affects your academic experience and plan to seek accommodations, it is your responsibility to inform Disability Support Services as soon as possible. Disability Support Services is located in the Learning and Student Development Center (LSDC) in Main Hall. Bridgette Dawson is the ADA representative; she can be reached at (304) 336-8216 or by email at bdawson@westliberty.edu. It is important to request accommodations early enough to provide adequate time to facilitate your request and provide faculty with written verification of eligibility.

Course Outline:

Chapter 1	Introduction to Computers and Python
Chapter 2	Variables and Experssions
Chapter 3	Types
Chapter 4	Branching
Chapter 4.5	Graphics
Chapter 5	Loops
Chapter 6	Functions
Chapter 7	Strings
Chapter 8	Lists and Dictionaries
Chapter 9	Exceptions
Chapter 10	Files
Chapter 11	Recursion
Chapter 12	Classes

Cheating: Don't do it. Students are expected to adhere to the official Academic Dishonestly Policy as stated below:

> *Academic Dishonesty, in whatever form, belies the stated philosophy of WLU "to promote the development of the intellectual, cultural, social, physical, emotional, moral, and vocational capacities of all persons within its sphere of influence." Individuals who commit acts of academic dishonesty violate the principles, which support the search for knowledge and truth. The academic community has established appropriate penalties and disciplinary action for such behavior that can include being expelled from WLU.*

Students are expected to do their own work on homework, lab, and program assignments. Learning to program is an individual task; you are expected to implement the programming assignments on your own. One person taking any part of another person's work (with or without their permission) and claiming it as his or her own is plagiarism and will not be tolerated. If you would agree that "he/she and I worked together" on implementing a program, then we would consider it cheating. Discuss your design or algorithm or logic in general terms, but write your own design, your own test cases, your own code, and your own implementation. If you have any concerns, ask me before submitting your work.

Do not show your source code (implementation) to any other student. It may seem an easy way to "show them how it's done" or "help them understand the problem." It is a recipe for getting into trouble. It is a temptation to copy the other person's work without figuring out how to solve the problem.

It is just as dishonest to allow someone to represent your work as their own as to do the reverse. This also means you are responsible for making sure that your code does not accidentally fall into someone else's hands. Don't leave memory sticks or printouts in a lab; don't leave source code files on a hard drive somewhere. Be aware that files that you put on the local hard drive (C or D or E) in a computer lab on campus STAY there until they are deleted. They do NOT automatically go away when you log out! If someone else finds your code and turns it in, YOU are responsible too!

Do not post your code on the Internet. This is an open invitation for someone else in the class to copy it and turn it in as theirs!

If you get help from a person who is not in the class, be extremely careful. Do not take code from anyone! Make sure the help you get is using the material covered in this class. You can be penalized in this situation also. If you work with a tutor, make sure you understand what the tutor is telling you. If they just "transplant" code into your program, (meaning either they wrote it for you or they dictated while you typed) you are being cheated of the understanding you need to do the next program. This is also considered cheating.

If you cheat on an assignment, you are cheating yourself. You are not getting the experience of working the problem out on your own, and you are not really learning the knowledge you are supposed to get from the assignment. This means that you are that much further behind on the next assignment.

If you get stuck at a point in the assignment, that is when you ask for help from me. Googling the problem is NOT the best way to find a solution! All assignments are designed to use the material covered in class. If you cannot figure out how to solve the problem with that, then it shows that there is something you do not understand about that material. It is a red flag to start asking questions! If you find something on the net which "solves the problem" but you do NOT understand it, you have cheated yourself of the understanding you need.