Course Goals and Schedule Fall 2018
COMP 150-001 Programming and Computer Algebra Dr. Laura K. Gross

Please find the course description, goals, modules, learning outcomes, semester schedule, and weekly rhythm below. Note also the tips on getting started in the course.

Course Description

In this course, you will learn to program a computer in the language R, which is widely used in the data-analysis industry, as well as academia. You will also learn to use the mathematical software Sage, which is another useful open-source tool in your mathematics toolbox. *If you are considering a career as a computer programmer or a computer-science teacher at the secondary level, take COMP 151, which is an introductory programming course designed for computer-science majors.*

In the first part of the course, you will develop algorithms (precise methods) to solve problems. You will implement them by writing computer programs in the language R. **Programming** will involve combining input, output, and control structures, including via repetition and decision structures. In the second part of our course, you will exploit the **computer algebra** system (CAS) Sage to analyze and solve mathematical problems by doing arithmetic and calculus, defining and manipulating mathematical expressions, evaluating functions, solving equations, and creating graphs. You may also explore higher-level topics of your choosing.

Course Goals and Learning Outcomes

Course Goals

1. Explain what computer programming is. Explain why computer programming is important. Distinguish between using software and writing programs. Describe the science, art, and utility of computer programming.
2. Analyze problems, specify solutions, design algorithms (precise methods), and write, test, and revise computer programs in mathematics and computer science. Use programming to solve problems of your own, for example problems arising in other mathematics courses.
3. Use a computer-algebra system to aid in understanding and solving mathematics problems, including problems of your own choosing.
4. Communicate about programming and mathematics in presentations, conversation, writing comments in code, writing textbook explanations, and reading.

Modules and Learning Outcomes

Module 1 (Week 1): Computers, R

1. You will cultivate your sense of community by introducing yourself, identifying your goals for the course, and getting to know your professor and colleagues.
2. You will familiarize yourself with your course syllabus, assignments, and course structure by engaging in initial activities.
3. You will use R and RStudio in the lab and install them on your computer by downloading packages from websites.
4. You will examine the respective roles of hardware and software in a computing system.
5. You will identify what computer scientists and computational mathematicians study and the techniques that they use.
6. You will examine the basic design of a modern computer.
7. You will examine the form and function of computer programming languages.
8. You will experiment with R by using `print`, `for` and `=`.

Module 2 (Week 2): Small programs

1. You will learn to define a function with zero or one arguments using the R command `function`, a function name, and the **name** of an argument if applicable.
2. You will compile the function by hitting the source button in RStudio.
3. You will invoke the function using the function name and the **value** of the argument if applicable.
4. You will modify and invoke functions in the RStudio editor.
5. You will create a .R text file containing a function definition.
6. You will apply and execute the steps in the programming process according to a handout.
7. You will construct appropriate names (identifiers) and expressions (instructions) by reading, exploring, and applying rules.
8. You will write functions that require application of the material to date.

Module 3 (Week 3): Computation with numbers, text, and vectors

1. You will compute with numbers.
2. You will use built-in mathematical functions and a random-number generator `sample` in R.
3. You will perform operations on strings.
4. You will define vectors.

5. You will identify the type of data as integer, (double-precision) floating point, or string, known to R as integer, double, or character, respectively, using commands such as typeof and is.double.
6. You will convert from one type to another using commands like as.double.
7. You will analyze problems, specify solutions, design algorithms, and write, test, and revise code, applying the material learned to date. The file you upload to Blackboard will contain function definitions and no invocation statements.

Module 4 (Week 4): Larger programs

1. You will learn to define a function with an appropriate number of arguments using the R command function, a function name, and the **names** of the arguments.
2. You will invoke the function by using the function name and the **values** of the arguments.
3. You will write programs that consist of functions that invoke other functions.
4. You will reduce code duplication and increase program modularity.
5. You will write functions that return values.
6. You will identify the scope of variables as local to the functions in which they are defined.
7. You will analyze problems, specify solutions, design algorithms, and write, test, and revise code, applying the material learned to date. The file you upload to Blackboard will contain function definitions and no invocation statements.

Module 5 (Weeks 5 and 6): Decision structures

1. You will identify the true/false Boolean data type, known to R as logical.
2. You will form simple true/false conditions using relational operators <, <=, ==, >, and >=.
3. You will form decision structures using if and if-else constructions to write programs that behave in different ways under different circumstances.
4. You will form conditions that use logical connectives &, |, and ! (*and*, *or*, and *not*, respectively, which are represented by the *ampersand*, *vertical bar*, and *exclamation point*, respectively).
5. You will analyze problems, specify solutions, design algorithms, and write, test, and revise code, applying the material learned to date. The file you upload to Blackboard will contain function definitions and no invocation statements.

Module 6 (Week 7): Repetition

1. You will revisit for loops and write cumulative loops by incrementing the value of a variable within a loop.
2. You will create nested loops by placing a loop inside a loop.

3. You will write indefinite loops using while, including initialization, a stop condition, and an update step.
4. You will analyze problems, specify solutions, design algorithms, and write, test, and revise code, applying the material learned to date. The file you upload to Blackboard will contain function definitions and no invocation statements.

Module 7 (Week 8): Recursion

1. You will identify recursive actions---actions that invoke themselves.
2. You will write functions that call themselves.
3. You will include a stop or go condition that is checked before a function calls itself.
4. You will identify errors in recursive functions.
5. You will analyze problems, specify solutions, design algorithms, and write, test, and revise code, applying the material learned to date. The file you upload to Blackboard will contain function definitions and no invocation statements.

Module 8 (Week 9): Transition to a computer algebra system (Sage)

1. You will synthesize the modules to date.
2. You will use the Sage Cell Server, which is somewhat like a calculator screen that lets you view output and type instructions.

Module 9 (Week 10): Calculation and constants in a computer algebra system (Sage)

1. You will use Sage to perform mathematical calculations, including those that involve built-in constants, such as pi and e.
2. You will assign values to variables in Sage.
3. You will distinguish between exact answers and numerical approximations.
4. You will use built-in functions in Sage, such as sin, cos, exp, and ln, and log.
5. You will set up an account at CoCalc and modify and create “worksheets,” including comments.

Module 10 (Week 11): Expressions, functions, and equations in Sage

1. You will define and use functions, expressions, and equations in Sage and distinguish among them.
2. You will find exact solutions to equations in Sage.
3. You will evaluate expressions in Sage, as well as distinguish between evaluating expressions and solving equations.

Module 11 (Week 12): Graphing in Sage

1. You will solve equations numerically in Sage.
2. You will graph functions in Sage, including graphing two functions on the same set axes, and scale appropriately.

Getting Started and Help Resources on the Blackboard course site

1. Electronic materials at BSU
2. Student Resources for Online Learning site
3. Personal success strategies

Course Schedule

Course Weeks Starting Weds	**Topics**	**Activities that support the module learning outcomes above**
Week 1 (Sep 5)	Computers, R	Goal identification, class surveys, personal introductions, software installation/use, exploration, syllabus reading, reading **Sep 10: Professor absent, classroom attendance optional**
Week 2 (Sep 12)	Small programs	Explorations, mini-lectures, exercise, student presentations, assignment
Week 3 (Sep 19)	Computation with numbers, text, vectors, and matrices	**Sep 19: Professor absent, classroom attendance optional** Assignment, explorations, mini-lecture
Week 4 (Sep 26)	Larger programs	Explorations, mini-lectures, exercises, student presentations, assignment
Week 5 (Oct 3)	Decision structures	Exploration, mini-lecture, exercise **Oct 8: No class (Columbus Day)**
Week 6 (Oct 10)	Decision structures	**Oct 10: Quiz 1** Exploration, mini-lecture, exercise, student presentations, assignment
Week 7 (Oct 17)	Repetition	Explorations, mini-lectures, exercises, student presentations, assignment
Week 8 (Oct 24)	Recursion	Explorations, mini-lectures, exercises, student presentations, assignment, **R application assignment plan**
Week 9 (October 31)	Synthesis/Transition	Mini-lecture, reading, exercise, question formulation **Nov 5: Quiz 2**
Week 10 (Nov 7)	Calculation and constants in Sage	Exploration, exercise, **R application assignment progress report**
Week 11 (Nov 14)	Mathematical	Exploration, exercises, student

Course Weeks Starting Weds	**Topics**	**Activities that support the module learning outcomes above**
	expressions, functions, and equations in Sage	presentations, **R application assignment**
Week 12 (Nov 21)	Graphing in Sage	Explorations, exercises, **assignment, Sage application assignment plan**
Week 13 (Nov 28)	Synthesis	Exploration, exercises, student presentations
Week 14 (Dec 5)	Applications	**Dec 5: Quiz 3** **Dec 10: Student presentations**
Week 15 (Dec 12)	Applications	**Dec 12: Student presentations,** question formulation **Dec 12: Last day of class**
Final Exam (Dec 17, 11:00-1:00)	R, Sage	Write a program; complete a Sage worksheet. Upload your .R file in the lab, and prepare your Sage worksheet for collection at the Cocalc site.

A Typical Week

In this course, each week begins on a Wednesday at 1:50 p.m. when class meets and ends the following Wednesday before class meets.
To succeed in the course, please:

- Review the weekly Blackboard item that summarizes the previous week's activities and provides an overview of the week ahead.
- Attend class, and focus closely on class activities.
- Review all assigned readings and explorations by Monday's class.
- Have a draft of the week's assignment completed by Monday's class.
- Submit your assignment on Wednesday.
- As you start the new week, review your submitted assignment and be prepared to present your completed work the following Monday.