Title: **TCCN:**	**"C" Programming** **COSC 1320**
Draft Course Description	Introduces the fundamental concepts of structured programming in the C language. Topics include data types; control structures; functions, structures, arrays, pointers, pointer arithmetic, unions, and files; the mechanics of running, testing, and debugging programs; introduction to programming; and introduction to the historical and social context of computing. Prerequisite: None
Draft Course Outcomes	Upon successful completion of this course, students will: 1. Analyze and explain the behavior of simple programs involving the fundamental programming constructs. 2. Modify and expand short programs that use standard conditional and iterative control structures and functions; choose appropriate conditional and iteration constructs for a given programming task. 3. 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. 4. Apply the techniques of structured (functional) decomposition to break a program into smaller pieces. 5. Describe the mechanics of parameter passing and demonstrate the difference between call-by-value and call-by-reference parameter passing. 6. Discuss the importance of algorithms in the problem-solving process, identify the necessary properties of good algorithms, and create algorithms for solving simple problems. 7. Use pseudocode or a programming language to implement, test, and debug algorithms for solving simple problems. 8. Discuss the representation and use of primitive data types and built-in data structures. 9. Explain the reasons for using different formats to represent numerical data. 10. Explain basic concepts of secure programming functions. 11. Discuss the properties of good software design. 12. Describe the phases of program translation from source code to executable code and the files produced by these phases; explain the software life cycle and its phases, including the deliverables that are produced. 13. Identify and describe the properties of a variable such as its associated address, value, scope, persistence, and size. 14. Explain how abstraction mechanisms support the creation of reusable software components.

Already in ACGM, course description has been revised, and learning outcomes have been added.