Revised 8/22

NOVA COLLEGE-WIDE COURSE CONTENT SUMMARY
CSC 221 – INTRODUCTION TO PROBLEM SOLVING AND PROGRAMMING (3 CR.)

Course Description

Introduces problem solving and implementation of solutions using a high-level programming language in a structured programming environment. Includes concepts and practice of structured programming, problem-solving, top-down design of algorithms, a high-level programming language syntax, control structures, arrays, and an introduction into object-oriented programming. First course in a three-course sequence. (CSC 221-222-223) The assignments in this course require mathematical problem-solving skills, algebraic modeling and functions, and use of variables. Lecture 3 hours per week.

General Course Purpose

CSC 221, CSC 222, and CSC 223 comprise the standard sequence of minimal programming content for computer science majors. The course sequence will teach the students to use high-level languages and their applications to problem solving by using algorithms within procedural and object-oriented techniques, while ensuring data adheres to a structured model. This course is the first course in the sequence. It introduces computer-based problem solving and implementation of solutions in a high-level programming language. Python is the preferred language for this course, institutions may offer using a different language to align with primary 4-year partner requirements.

Course Prerequisites/Corequisites

None.

Course Objectives

Upon completing the course, the student will be able to:

Civic Engagement

- Engage and build technology that responds to human needs and helps people navigate institutional systems

Critical Thinking

- Assess why certain solutions might not work and to save time in coming up with a more efficient approach

Professional Readiness

- Work well with others and display situationally and culturally appropriate demeanor and behavior

Quantitative Literacy

- Perform accurate calculations, interpret quantitative information, apply and analyze relevant numerical data, and use results to support conclusions

Scientific Literacy

- Represent real-world objects and processes virtually by identifying properties, behavior, and operations relevant to solving problems on a computer.

Written Communication

- Develop, convey, and exchange ideas in writing, as appropriate to a given context and audience

Basic concepts of computer systems

- Differentiate computer components by functionality.

- Define basics of computer storage devices.
- Illustrate the computer structure.
- Define Binary and Hexadecimal numeration systems.
- Define types of software.
- Explain the use of computers, and the social impact they have.
- Discuss secure programming
- Evaluate the ethical aspects of programming

Processing Code

- Editors, compilers and/or interpreters; distinguishing source code, object code, and executables.
- Reading and evaluate compilation error messages.
- Executing programs.
- Analyzing and resolving run-time errors.

Problem analysis and algorithmic modeling

- List and apply the steps involved in problem solving through algorithmic modeling.
- Describe activities related to program modeling and design including algorithm development.
- Solve problems using techniques such as pseudocode, flowcharts, and model development.
- Verify algorithms and identify errors.
- Distinguish between procedural techniques and object-oriented techniques.
- Write programs using good programming practices.

Use of data

- Compare and contrast data types.
- Describe the use of variables.
- Build expressions using variables, literal data, and operators, correctly using rules of operator precedence.

Decision structures

- Describe how conditional selection operations are used to alter the sequential execution of a program.
- Describe how relational and Boolean operators are used to form logical expressions that evaluate to true or false
- Identify techniques to evaluate selection statements for logic errors.
- Develop programs using sequential and selection operations.

Repetition structures

- Describe how repetition structures are used to alter the sequential execution of a program.
- Choose appropriate repetition structures based on the type of application.
- Identify techniques to evaluate repetition statements for logic errors.
- Develop programs using repetition structures.

Programming with Procedures

- Apply modularization to manage complexity of programming
- Describe the roles of parameters in a procedure definition.
- Illustrate parameter passing when invoking procedures.
- Solve problems using procedures.

Classes and Introduction to Libraries

- Describe information hiding and encapsulation.
- Describe the concept of class and object of a class.
- Use language classes from the standard library to develop programs.

Arrays

- Define the nature and purpose of an array.
- Use arrays as parameters and returned values in procedures.
- Evaluate programs that use arrays.
- Develop applications using arrays.

Major Topics to be Included

- Basic concepts of computer systems
- Processing Code
- Problem analysis and algorithmic modeling
- Use of data
- Decision structures
- Repetition structures
- Programming with Procedures
- Classes and Introduction to Libraries
- Arrays