Computer Science Department
cs.salemstate.edu

CSC 279 C + C++ **4 cr.**

Instructor: TBA **Office:** location **Phone:** (978) 542-extension
email: TBA@salemstate.edu **Office Hours:** days and times

Section	Time	Room	Final Exam
nn	days and times	location	Date and time

Catalog description:

This course presents the particular goals, features, and strengths and limitations of the C and C++ programming languages. C's capabilities and limitations as a procedural programming language are examined, followed by an exploration of C++ as an object-oriented language that provides access to C's feature set. Topics include language grammar rules and their effect on programming style, operators, pointer and reference types, bit manipulation, memory management, and the utilization of the STL (Standard Template Library). Programming assignments will highlight the use of each language in appropriate contexts (e.g. C: systems programming, text processing; C++: program-solving strategies emphasizing OO and the use of the STL). Fundamental programming language paradigms, type systems, and memory allocation and management strategies are presented and discussed, followed by comparative analysis of the languages utilized in this course and its prerequisites. Four lecture hours per week, plus extensive programming work outside of class.

Prerequisite: CSC115 or CSC 202J.

Goals:

CG01: to present typical concepts and features of a procedural programming language (C);
CG02: to provide additional experience in problem-solving and programming in an object-oriented programming language (C++);
CG03: to enhance students' skills in problem analysis and program design and implementation via the use of C and C++ capabilities and their related toolkits;
CG04: gain a basic level of understanding of fundamental programming language concepts.

Objectives:

Upon completion of this course, the student will have demonstrated the ability to:

CO01: understand and utilize the syntax and special capabilities of the C and C++ languages, including preprocessors, header files, pointer vs. reference in each language, operators, bit manipulation, and memory management;
CO02: determine whether to select or create algorithms and language features for the solution of a complex problem, and use these ingredients effectively to generate a solution to the problem;
CO03: solve problems that appropriately utilize the features of the C language;
CO04: solve problems that appropriately utilize the features of the C++ language, including the various types of reuse available via object-oriented programming;
CO05: understand and use a variety of components from the C++ Standard Library and Standard Template Library;
CO06: design and implement solutions to relatively large-scale problems using object-oriented tools, and provide appropriate documentation for the solutions;
CO07: demonstrate knowledge of fundamental programming language paradigms, type systems, and memory allocation and management strategies.

Program Outcome vs. Course Objectives matrix

Program Objective (condensed form)	**CO01**	**CO02**	**CO03**	**CO04**	**CO05**	**CO06**	**CO07**
PO-A: apply knowledge of computing and math	✓	✓	✓	✓	✓	✓	
PO-B: analyze a problem and define its computing requirements		✓	✓	✓	✓	✓	✓
PO-C: design, implement and evaluate applications		✓	✓	✓	✓	✓	

Program Objective (condensed form)	CO01	CO02	CO03	CO04	CO05	CO06	CO07
PO-D: function effectively in teams to accomplish a common goal							
PO-E: professional, ethical, and social responsibilities							
PO-F: communicate effectively with a range of audiences						✓	
PO-G: local and global impact of computing on people and society							
PO-H: need for continuing professional development							
PO-I: use current techniques, skills, and tools	✓	✓	✓	✓	✓	✓	✓
PO-J: apply theory and principles to model and design systems		✓	✓	✓	✓	✓	✓
PO-K: apply design and development principles in constructing software		✓				✓	
note - full statements of the Program Outcomes (program objectives) for the Computer Science Major can be found in the document *Computer Science Major Program Educational Objectives and Program Outcomes* on the Assessment page of the Computer Science Major (cs.salemstate.edu)							

Topics:

- programming language concepts **PL1(1), PL2(1), PL6(2), PL14(2), SDF2(2)**
  - programming paradigms
    - procedural
    - object-oriented
    - functional
    - declarative
    - multi-paradigm
    - emerging and/or specialized paradigms
  - type systems
    - "strongly-typed" vs "weakly-typed" vs. "type safe"
    - static (compile-time) type checking
    - dynamic (run-time) type checking
  - memory allocation and management
    - static vs. dynamic
    - direct vs. indirect
    - stack-based
    - heap-based
- the C programming language: **AR2(1), PL2(2), PL4(1), PL6(2), SDF2(2)**
  - the language environment
    - the standard C library
    - header files
    - the C preprocessor
  - the language syntax and features
    - data types, type conversions
    - input/output, files and streams
    - references vs. pointers
    - strings, structures, unions
- the C++ programming language and problem solving in C++ **PL1(3), PL4(1), PL5(1), PL6(1), PL10(1), PL14(2), SDF3(2), SDF4(3)**
  - review of object-oriented concepts
    - behavior and state
    - instances and classes
    - interface and implementation
  - reuse
    - composition, aggregation
    - inheritance, templates
  - creating software components
    - classes and objects

        - streams
        - containers and iterators
    - use of C++ Standard Library and the STL
- applications with the use of C/C++ **AL1(1), AL2(1), PL7(1), SDF1(1), SE3(1)**
    - review of standards for algorithm design
    - problem solving and problem-solving strategies
        - top-down, divide and conquer strategies
        - breadth first and backtracking algorithms

Programming Assignments: There will be 4 to 8 substantial programming assignments (possibly including one large project divided into progressive stages) in which students will be required to implement appropriate design components. The following provide a list of typical topics for the programming assignments:

- Projects emphasizing the procedural programming paradigm (topics chosen from, but not limited to, the following):
    - data compression
    - data encryption
    - graph algorithms
    - numeric applications (including high-precision arithmetic)
    - rational number and complex number classes
    - searching and sorting
    - string processing
- Projects emphasizing the OO programming paradigm (topics chosen from, but not limited to, the following):
    - bit vectors and applications
    - knight's tour problem
    - large precision integers
    - rational number and complex number classes
    - strings and string processing; pattern matching

All programs must conform to departmental guidelines for design and implementation, and laboratory reports must conform to the written guidelines supplied by the instructor.

Programming exercises: There will be short programming and design exercises to be completed outside the class. The exercises will concentrate on language syntax, programming style, and familiarization with the contents of the C++ Standard Library and Standard Template Library

Exams and quizzes: There will be one mid-term exam and a comprehensive written two-hour final examination.

The course grade will be determined using the following approximate weights: laboratory exercises (done outside the class) - 20%, programming assignments – 40%, examinations (midterm and final) -30%, written homework - 10%.

Course Objective / Assessment Mechanism matrix

	Written Homework	Programming Exercises	Programming Projects	Examinations
CO01	✓			✓
CO02	✓			✓
CO03			✓	✓
CO04	✓	✓	✓	✓
CO05	✓	✓	✓	✓
CO06			✓	
CO07		✓	✓	✓

Bibliography:

Bronson, Gary. **A First Book of ANSI C**. **Fourth Edition.** Cengage Learning, 2006.
Collopy, David M. **Introduction to C Programming: A Modular Approach**. Prentice Hall, 2002.
Cormen, Thomas H.; Leiserson, Charles H.; Rivest, Ronald L.; Stein, Clifford. **Introduction to Algorithms**. **Third Edition.**

The MIT Press, 2009.
Deitel, Harvey M.; Deitel. Paul J. **C++ How to Program: Early Objects Version. Tenth Edition.** Prentice Hall, 2016.
Deitel, Paul J.; Deitel, Harvey M. **C: How to Program**. **Eighth Edition.** Prentice Hall, 2015.
Fowler, Martin; with Scott, Kendall. **UML Distilled: A Brief Guide to the Standard Object Modeling Language**. **Third Edition.** Addison-Wesley, 2003.
Friedman, Frank L.; Koffman, Elliot B. **Problem Solving, Abstraction, and Design using C++. Sixth Edition.** Addison-Wesley, 2011.
Gaddis, Tony**;** Walters, Judy; Muganda, Godfrey. **Starting Out with C++: Early Objects. Ninth Edition.** Addison-Wesley, 2016.
Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John. **Design Patterns: Elements of Reusable Object-Oriented Software**. Addison-Wesley, 2015.
Horowitz, Ellis; Sahni, Sartaj; Rajasekaran, Sanguthevar. **Computer Algorithms/C++**. **Second Edition.** Silicon Press, 2008.
Horstmann, Cay S. **C++ for Everyone. Third Edition.** Wiley, 2017.
Horstmann, Cay S.; Budd Timothy A. **Big C++.** Wiley, 2015.
Kernighan, Brian W.; Ritchie, Dennis M. **The C Programming Language. Second Edition.** Prentice Hall, 1988.
Koffman, Elliot B.; Hanly, Jeri R. **Problem Solving and Program Design in C. Eighth Edition**. Addison-Wesley, 2015.
Loudon, Kyle. **Mastering Algorithms with C**. O'Reilly, 1999.
Main, Michael; Savitch, Walter. **Data Structures and Other Objects Using C++**. **Fourth Edition.** Addison-Wesley, 2011.
Musser, David R.; Saini, Atul; Derge, Gillmer J. **STL Tutorial and Reference Guide: C++ Programming with the Standard Template Library**. Addison-Wesley, 2009.
Oestereich, Bernd. **Developing Software with UML: Object-Oriented Analysis and Design in Practice. Second Edition.** Addison-Wesley, 2002.
Prata, Stephen. **C++ Primer Plus. Sixth Edition.** Sams, 2011.
Prata, Stephen **C Primer Plus, Sixth Edition**., Sams, 2013.
Richard Neapolitan. **Foundations of Algorithms, Fifth Edition**. Jones & Bartlett Learning, 2014.
Savitch, Walter J. **Problem Solving with C++. Ninth Edition.** Addison-Wesley, 2014.
Schildt , Herbert. **C++ Programming Cookbook**. McGraw-Hill, 2008.
Schildt , Herbert. C++. **The Complete Reference, 4th Edition.** McGraw-Hill, 2008.
Schildt, Herbert. **C/C++ Programmer's Reference. Second Edition.** McGraw-Hill, 2000.
Schildt, Herbert. **C: The Complete Reference**. **Fourth Edition.** Osborne/McGraw-Hill, 2002.
Shalloway, Alan; Trott, James R. **Design Patterns Explained: A New Perspective on Object-Oriented Design. Second Edition.** Addison-Wesley, 2004.
Seacord, Robert. **Secure Coding in C and C++. Second Edition**. Addison-Wesley, 2013.
Sedgewick, Robert. **Algorithms in C**. **Third Edition. Parts 1-4: Fundamentals, Data Structures, Sorting, Searching. Third Edition.** Addison-Wesley, 1998.
Sedgewick, Robert. **Algorithms in C, Part 5: Graph Algorithms. Third Edition.** Addison-Wesley, 2002.
Weiss, Mark Allen. **Data Structures and Algorithm Analysis in C++. Fouth Edition.** Addison-Wesley, 2013.

Academic Integrity Statement:

"Salem State University assumes that all students come to the University with serious educational intent and expects them to be mature, responsible individuals who will exhibit high standards of honesty and personal conduct in their academic life. All forms of academic dishonesty are considered to be serious offences against the University community. The University will apply sanctions when student conduct interferes with the University primary responsibility of ensuring its educational objectives." Consult the University catalog for further details on Academic Integrity Regulations and, in particular, the University definition of academic dishonesty.

The Academic Integrity Policy and Regulations can be found in the University Catalog and on the University website (http://catalog.salemstate.edu/content.php?catoid=13&navoid=1295#Academic_Integrity). The formal regulations are extensive and detailed - familiarize yourself with them if you have not previously done so. A concise summary of and direct quote from the regulations: "Materials (written or otherwise) submitted to fulfill academic requirements must represent a student's own efforts". *Submission of other's work as one's own without proper attribution is in direct violation of the University's Policy* and will be dealt with according to the University's formal Procedures. *Copying without attribution is considered cheating in an academic environment - simply put,* ***do not do it!***

University-Declared Critical Emergency Statement:

In the event of a university-declared emergency, Salem State University reserves the right to alter this course plan. Students should refer to www.salemstate.edu for further information and updates. The course attendance policy stays in effect until there is a university-declared critical emergency.

In the event of an emergency, please refer to the alternative educational plans for this course, which will be distributed via

standing class communication protocols. Students should review the plans and act accordingly. Any required material that may be necessary will have been previously distributed to students electronically or will be made available as needed via email and/or Internet access.

Equal Access Statement:

"Salem State University is committed to providing equal access to the educational experience for all students in compliance with Section 504 of The Rehabilitation Act and The Americans with Disabilities Act and to providing all reasonable academic accommodations, aids and adjustments. **Any student who has a documented disability requiring an accommodation, aid or adjustment should speak with the instructor immediately.** Students with Disabilities who have not previously done so should provide documentation to and schedule an appointment with the Office for Students with Disabilities and obtain appropriate services."

Note: This syllabus represents the intended structure of the course for the semester. If changes are necessary, students will be notified in writing and via email.