WICHITA STATE UNIVERSITY

CS 311: Object Oriented Programming
Spring 2018

Instructor:	Joe Shobe
Classroom; Days/Time:	Jabara 128; Tuesday & Thursday, 4:05-5:20 PM
Office Location/Hours:	Jabara 205A; 3:00-4:00pm Tue/Thu and by appointment via e-mail
Email:	**joe.shobe@wichita.edu**
Credit Hours:	4.0
Prerequisites:	CS 211: Introduction to Programming

How to use this syllabus
This syllabus provides you with information specific to this course, and it also provides information about important university policies. This document should be viewed as a course overview; it is not a contract and is subject to change as the semester evolves.

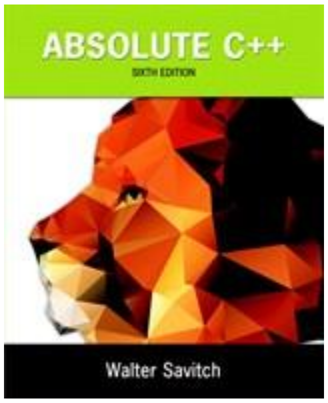
Course Description
The main objective is to introduce the concepts of object-oriented programming. The expected topics include data abstraction, classes and objects, methods, inheritance, polymorphic variables, dynamically-bound method calls, and data encapsulation. Gives programming experience in an object-oriented programming language, i.e. *C++*.

Course Goals and Student Learning Objectives
Upon successful completion of this course, students will have demonstrated the following abilities:

- The knowledge to use the basic elements of object oriented programming.
- The ability to develop algorithms to solve simple application problems.
- The capability of transforming algorithms into object oriented programs.
- The ability to understand and correct syntax errors generated by a compiler.
- The ability to diagnose and correct logic errors in computer programs using an interactive debugger.
- Working knowledge of good programming practices.
- Working knowledge of programming in *C++*.

Textbook
Pearson: Absolute C++, Sixth Edition by Walter Savitch
ISBN-10: 0-13-397078-7 ISBN-13: 978-0-13-397078-4

6th Edition



ISBN-10: 0-13-397078-7
ISBN-13: 978-0-13-397078-4

Class Protocol

- The primary method of instruction will be through class lectures and in class discussions. Students can utilize their personal computers or workstations located in the computer labs for the assigned programming projects.
- The daily lecture and activity schedule will be posted on the course web page and in this syllabus (also stored on the course web page). It is the responsibility of the students to monitor this resource regularly. Course material, assignment, and exam related deadlines will be posted there.
- Attendance in lectures and other class activities (lecture periods, tests, examinations, or other scheduled meetings) is required of every student. The attendance record begins with the first meeting of the class, and one who registers late is responsible for class work missed.
- It is the responsibility of the students to make up for the missed classes.
- Students are highly encouraged to raise and discuss with the instructor any problem/difficulty faced in the class, including understanding the covered material.
- Course related updates and modification would be typically posted on the course web page.
- Please turn off sound on phones, laptops, PDAs, watches, and any other noise making devices.
- Students may use a calculator for exams, but the calculator must be a typical calculator that does not include functionality beyond performing mathematical calculations (devices such as smartphones, tablets, watches, PDAs, laptops, etc are not allowed).

Grading Scale

Students will be evaluated, and assigned final grades, based on their performance on examinations and assignments, as follows:

- Exams: 50%
- Assignments/Programming Projects: 50%

WSU uses a +/- grading scale for final grades and to calculate grade point averages. In this class, grades are assigned according to the following distribution:

Percentages	Letter Grade	Grade Points	Interpretation
[93, 100]	A	4.00	*The A range denotes excellent performance.*
[90,93)	A-	3.70	
[87,90)	B+	3.30	
[83,87)	B	3.00	*The B range denotes good performance.*
[80,83)	B-	2.70	
[77,80)	C+	2.30	
[73,77)	C	2.00	*The C range denotes satisfactory performance.*
[70,73)	C-	1.70	
[67,70)	D+	1.30	
[63,67)	D	1.00	*The D range denotes unsatisfactory performance.*
[60,63)	D-	0.70	
[0,60)	F	0.00	*F denotes failing performance.*

Note: Other classes might assign grades differently; therefore, be sure to understand the different grading scales in all of your classes. In the above table, a square-bracket directly before or after the percentage value implies *inclusiveness* while a parenthesis after the percentage value implies *exclusiveness*. The percentage [90,93) means a grade value that is greater than or equal to 90 and less than 93 (inclusive of 90, exclusive of 93).

Assignments

There will be approximately seven programming assignments. Each assignment's description will be posted on the course web page and are generally due on a Saturday, 11:59pm. Refer to the class schedule at the end of this syllabus for specific due dates.

Late Assignments

All assignments must be turned in by midnight of the due dates for full credit and allowed to be turned in up to 3 days late, but will be penalized 10% per day, scoring a 0 thereafter. Consideration of a late assignment and the associated penalty is completely at the instructor's discretion. Students are well advised not to engage in late assignment practices.

Exams

There will be approximately two exams, a midterm and a final. All exams must be attempted and consist of a test conducted during the regular class hours.

Missed Exams

No makeup exams; exception can be made in case of emergency or documented illness. In such situations, the involved student must inform the instructor at the first available chance; however, the final decision is at the discretion of the instructor.

Development and Submissions

Completion of assignments requires a prior understanding of using WSU's cslab Linux environment, Linux servers, and a few Linux development tools (such as the command-line, a text editor and compiler) which are not covered in detail by this class. For a tutorial on using the *cslab Linux Environment*, refer to the *EECS Tutorial* document found on the course web page. Source and other required files for assignments/projects are submitted via the procedure outlined within the *Linux Server* document (also on the course web page) and later graded by compiling (using g++, the GNU GCC compiler for C++) and testing the resulting executable(s) on the same Linux server. Students are strongly encouraged to use more sophisticated development tools (such as the Eclipse IDE C/C++ for Developers) for developing and testing assignments/projects. However, the compatibility and submission of files must still comply with the grading expectations already set forth.

UNIVERSITY POLICIES

General

Some general university policies pertaining to all syllabi can be found at:
https://webs.wichita.edu/?u=ofdss&p=/students/syllabusinformation

Academic Honesty

Students are responsible for knowing and following:

- Student Code of Conduct policy http://webs.wichita.edu/inaudit/ch8_05.htm
- Student Academic Honesty policy http://webs.wichita.edu/inaudit/ch2_17.htm

The following rules will apply in this course:

- Feel free to discuss programming assignments in this course with others, including other current or previous students in this course, tutors, or TAs, but the work you turn in must be your own work and done by you.
- For assignments:
  - All students must write their own programs individually.
  - No student is to give a copy of their program to another student of the course, or allow another student to copy their work.
  - No student should make or use a copy of another student's programs or work.
  - If you have any questions, then ask the instructor.
- Failure to follow the above rules for assignments will be considered plagiarism or cheating. Likewise, any copying or use of unpermitted reference material during examinations will be considered cheating.
- Plagiarism and cheating are very serious offenses and will not be tolerated. These cases will reported to the proper university authorities and will result in grading penalties including an assigned grade of zero for work plagiarized/cheated on, and for a second offense an F for the course.

Definition of a Credit Hour

A "credit hour" is a measure of graduate or undergraduate academic work represented in intended learning outcomes and verified by evidence of student achievement that reasonably approximates not less than one hour of classroom or direct faculty instruction and a minimum of two hours of out-of-class student work for each week of instructional time for approximately fifteen weeks for one semester, or an equivalent amount of work over a different amount of time.

Success in this 4.0 credit hour course is based on the expectation that students will spend approximately 2½ hours for instruction and another 6¾ to 9 hours for preparation/studying/programming or other course related activities each week for a total of about 150 to 180 hours for the semester.

Tentative Class Schedule
CS 311, Object Oriented Programming, Spring 2018

Week	Date	Topics
1	Tue, Jan 16	Syllabus, Introduction, Review, Programming Styles, Eclipse IDE
	Thu, Jan 18	Chap 6.1: Review – Structures Chap 6.2: Classes
	Sat, Jan 20	Assignment 1: Chap 6; due by 11:59pm
2	Tue, Jan 23	Chap 7.1: Constructors Chap 7.2: More Tools
	Thu, Jan 25	Chap 7.3: Vectors
	Sat, Jan 27	Assignment 2: Chap 6-7; due by 11:59pm
3	Tue, Jan 30	Chap 11.1: Separate Compilation Chap 11.2: Namespaces
	Thu, Feb 1	Chap 8.1: Basic Operator Overloading
4	Tue, Feb 6	Chap 8.2: Friend Function and Automatic Type Conversion
	Thu, Feb 8	Chap 8.3: References and More Overloading Operators
	Sat, Feb 10	Assignment 3: Chap 6-8, 11; due by 11:59pm
5	Tue, Feb 13	Chap 9.1: Array Type for Strings Chap 9.2: Character Manipulation Tools
	Thu, Feb 15	Chap 9.3: The Standard Class *string*
6	Tue, Feb 20	Chap 10.1: Review – Pointers Chap 10.2: Review – Dynamic Arrays
	Thu, Feb 22	Chap 10.3: Classes, Pointers, and Dynamic Arrays
	Sat, Feb 24	Assignment 4: Chap 6-11; due by 11:59pm
7	Tue, Feb 27	Chap 14.1: Inheritance Basics
	Thu, Mar 1	Chap 14.2: Programming with Inheritance
8	Tue, Mar 6	Chap 15.1: Polymorphism and Virtual Functions
	Thu, Mar 8	Chap 15.2: Pointers and Virtual Functions
	Sat, Mar 10	Assignment 5: Chap 6-11, 14-15, due by 11:59pm
9	Tue, Mar 13	Chap 12.1: Review – I/O Streams Chap 12.2: Review – Tools for Stream I/O
	Thu, Mar 15	Mid-Term Exam: Chap 6-11, 14-15
10	Tue, Mar 20	Spring break (no class)
	Thu, Mar 22	Spring break (no class)
11	Tue, Mar 27	Chap 12.3: Stream Hierarchies Chap 12.4: Random Access to Files
	Thu, Mar 29	Chap 18.1: Exception Handling Basics Chap 18.2: Programming Techniques for Exception Handling
	Sat, Mar 31	Assignment 6: Chap 6-12, 14-15, 18; due by 11:59pm
12	Tue, Apr 3	Chap 16.1: Function Templates
	Thu, Apr 5	Chap 16.2: Class Templates
13	Tue, Apr 10	Chap 16.3: Templates and Inheritance
	Thu, Apr 12	Chap 17.1: Review – Nodes and Linked Lists Chap 17.2: Review – Linked List Applications
14	Tue, Apr 17	Chap 17.3: Iterators Chap 17.4: Trees
	Thu, Apr 19	No class
	Sat, Apr 21	Assignment 7: Chap 6-12, 14-18; due by 11:59pm
15	Tue, Apr 24	Chap 19.1 Iterators
	Thu, Apr 26	Chap 19.2 Containers
16	Tue, May 1	Chap 19.3 Generic Algorithms
	Thu, May 3	Final Exam Review
Final	Tue, May 8	Final Exam: Chap 6-12, 14-19