CS 132: Computer Science II C++

Winter 2018 - Shuksan Hall 233
Section A (Item# 2840) T, TH : 12:20 – 3 pm
Section S (Item# 2842) M, T, TH : 3:20 – 5:10 pm

Instructor: Lonnie Heinke
Office: Shuksan 114
Email: lheinke@everettcc.edu (preferred and you should get a response within 24hrs during the week)
Grades, assignments, class schedule and other class information will be available on Canvas.

Office Hours:
Wednesday: 2 – 4 pm in office (SHK 114) (Note: on days with Dept. Meeting 2 – 3:30pm)
Friday: 11am – 12:30 pm in office (SHK 114)
These are times that are specifically reserved for you to be able meet with me to talk about programming or course questions. If you are unable to make it to any of the posted office hours, please contact me to make an appointment for an alternative time.
SHK 233 Before/After class: M, T, TH 3-3:20 (Instructor or Student Assistant), M, T, TH 5:10-5:30pm, T, TH 12-12:20pm

Required Text/Materials:
Problem Solving with C++, 10th Edition by Walter Savitch (ISBN-13: 978-0134448282)
External storage for your programs (flash drive, cloud storage (i.e. Google Drive with your everettcc email account))

Course Aims:
You will gain the knowledge and ability to use the advance features of the C++ language emphasizing object-oriented concepts and fundamental data structure techniques and become *comfortable* and *confident* in using them to write programs. In addition, this course will prepare you for future studies in the advanced concepts of C++ and other subjects in the Computer Science or Engineering curriculum.

Prerequisites: CS& 131 with a grade of C or higher; or instructor permission

Course Outcomes (or what you will learn from this class):
- Demonstrate an understanding of pointers and basic memory management.
- Implement abstract data types, such as structures, and classes.
- Demonstrate an understanding of the object oriented nature of the C++ programming through the design, implementation and use of classes and associated concepts (e.g., inheritance, polymorphism, friend functions, and access modifiers).
- Properly employ basic searching, sorting, and recursion algorithms on data sets.
- Design, implement, and manipulate elementary data structures, e.g., linked lists, stacks, queues and binary trees, to fulfill specific design challenges.
- Demonstrate understanding of common computer science algorithmic approaches.

Supports Engineering/Computer Science Department Learning Outcomes:
Area #1: Demonstrate analytical problem solving skills: Assessed through exams, projects and homework that require students to write computer programs to solve a variety of problems.
Area #5: Apply engineering design process: Assessed by evaluating project deliverables and process documentation in one or more projects.

Supports EvCC Core Learning Outcomes:

Area #1: Engage and take responsibility as active learner: Introduced

Area #2: Think critically: Critical thinking is an integral skill to algorithm development and is assessed through exams, projects and homework assignments.

Area #5: Utilize information literacy skills: Introduced

Area #6: Demonstrate computer and technology proficiency: Assessed through exams, projects and homework assignments that require students to access Web-based resources, access online postings, and write, debug and test code using programming tools.

Grading Breakdown:

Exams	40%	Week 4, 7, finals week (Thursday, March 22nd)
Lecture Study Notes	5%	Roughly Bi-weekly
Quizzes	10%	Start of each class
Lab Work	5%	1-2 per week
Programs	40%	7-8 throughout semester

Your final grade will be based on the following breakdown:

A: 93% or higher	B+: 87% or higher	C+: 77% or higher	D+: 67% or higher	F: below 60%
A-: 90% or higher	B: 83% or higher	C: 73% or higher	D: 60% or higher	
	B-: 80% or higher	C-: 70% or higher		

Grading Note: You must have an Exam average and a Program average of 70% or better to pass the class. If you have an average in either category that is below 70%, then the best grade you can receive for the class is **C –** .

Assignments and Exams:

Programs:

- Programs are an essential part of the class, and so it is very important that you work on your program until it is in a running condition (can complete the majority of the requirements). If a program can't meet these conditions, it is better for a student to continue working with the program until it does.
- Programs are due at the beginning of class...before the class starts. If a program is turned in later than the start of class, there will be a 15% deduction. After the due date, a program will only be accepted within 2 additional days. These deadlines are to help you keep up with the current program, instead of continuing to work on a previous program.

Exams:

- There are three exams given during the quarter (weeks 4, 7 and Finals week).
- Each exam will **focus** on the topics covered in the 3-4 weeks before the exam, but you will still need to be able to use prior concepts.
- The exams will be short answer where you will need to explain or give the output for short sections of code, write short sections of code, or explain programming concepts.
- During the exam, you can use one 3x5 card (that's in inches) which includes your *hand written* reference notes of what you feel is important information. The cards must be handwritten and will be checked before the exam. Any reference card that is not the right size or is not believed to be hand written will not be allowed to be used during the exam.

Lecture Study Notes:
- After each lecture topic is finished, you will need to show your study notes of the main points from that topic. This could include definitions, brief code examples and possible pictures to help illustrate topics. *It is suggested that you work on your study notes after each class, as a way to help you think about the topics and prepare for the next quiz.*
-Lecture study notes will be graded based on completeness.

Quizzes:
- Will be given at the start of each class to help the student to review and reinforce some of the main topics from previous class. Quizzes also give the students a preview of what test questions may look like.
- Missed quizzes cannot be made up.
- 3 lowest quiz scores will be dropped.

Computer Usage:
To encourage active mental participation and note taking during the lectures. There will be **no** computer usage during the lecture other than note taking or viewing the lecture slides. **No programming is allowed during lectures, as well as printing.**

Cell Phone Usage:
Cell phones can only be used during official class break times. To learn, a person needs to be able to focus on a subject....you can't focus on programming while leveling up your kingdom, capturing a Pokémon, or texting your sweetheart. If you finish a programming exercise early, think about other things you could try or experiment with in regard to the topic. Programming exercises are just the beginning of learning a concept (the bare minimum)....push yourself and progress from minimum knowledge to elementary or intermediate knowledge.

More about Programming Assignments and Academic Honesty:
Start early on the programming assignments, because you will often encounter problems where you get stuck. By starting early, you will have time to walk away from the problem to talk with others, think, and do more reading. You will only have this "walk away" time if you are starting early.

I can't emphasize "*starting programming assignments early*" strongly enough. I believe that your success or failure in this class depends on it. In addition, many of the students from previous quarters also said the same thing when asked about words of advice for students entering the class.

I would like to strongly encourage you to collaborate and talk with others about your projects or problems. By discussing the programming issues, both parties will develop a deeper understanding of programming.

Good collaboration is discussing programming ideas/solutions or how a specific language feature works. Examples of bad collaboration are sitting down with a copy of someone's code and typing from their code, or if you are "helping" another student by telling them what to type in. In these examples of bad collaboration, the student is entering code, but is not really learning or understanding...they are not creating code. Creating code will be a struggle at times, but this is where true learning comes from.

Each person will have a different way of thinking about a problem, and so their code will be uniquely their own. If multiple students turn in code that is the same, then all the people who turned in duplicate code will receive a score of zero. Unless stated otherwise, all assignments are individual assignments ….which means you are to write your own code. Imagine if you were taking a piano class

but always had someone else practice/play for you. Would you be truly learning? Would you be able to improve in the craft?

Please do collaborate in thinking about assignments, and discuss problems or ideas with your classmates, but when you go to write your code, then do this aspect of the work on your own. Please talk with me if you are still unclear about the differences of collaborating and copying.

Course Schedule: *This schedule is an estimate of when topics will be covered. It is mostly a guide to what order topics will be covered and is subject to change as the quarter progresses.*

Week	Content	Other Information
Week 1	Course Intro, Recursion, Classes	Deadline for 100% refund (Friday, 4:30pm)
Week 2	Classes – Static things	Deadline to drop a class without record (Friday,4:30pm).
Week 3	Classes – operator overloading - exceptions Linked List data structure	Deadline for 50% refund (Friday); 20-40 Plan Payment deadline (Friday)
Week 4	Move Assignment/Copy	**Exam 1**: covers weeks 1-3
Week 5	Inheritance	Deadline for making changes to residency
Week 6	Polymorphism	20-40 Plan Payment deadline (Friday)
Week 7	Templates	**Exam #2:** covering weeks 4-7
Week 8	Data Structures: Tree, Stack, Queue	
Week 9	Sorting: Merge Sort	
Week 10	Misc Topics	
Finals Week		Thursday March 22nd Sect A: 12- 1:50, Sect S: 2- 3:50

Accommodations for students with disabilities:
I want to encourage students with disabilities (whether visible or not) to talk with me during my office hour if accommodations are needed to allow you to study effectively. Students are also encouraged to register with the Center for Disability Services office on campus to verify their eligibility and help facilitate appropriate recommendations for accommodation. Center for Disability Services is located in Parks 267 (across from the bookstore).

Small adjustments may need to be made to this syllabus due to the fluid nature of a class. Any changes will be announced in class, and on Canvas.