SYLLABUS
CS 202: Programming Systems
Fall 2020

Prerequisite:	**CS163 Data structures** *Students are expected to be able to fluently write complete programs in C++ at the linux command line (without an IDE) using classes, pointers, dynamic memory, developing data structures such as Linear Linked Lists, Circular Linked Lists, Doubly Linked Lists, Arrays of Linked Lists, Linked Lists of Arrays, Trees, and Graphs.*

Instructor:	**Karla Steinbrugge Fant**	
E-mail:	**karlaf@pdx.edu**	
Office hours:	**T/TH 2:30-3:30**	**Zoom ID: 257 779 553**
SLACK Channel:	**https://psu-ccu9286.slack.com**	
Live Streaming Sessions:	**T/TH 4:40-6:10**	
Homework: Recitation	**Monday 2-5:50** **Tuesday 2-5:50** **Wednesday 2-5:50** **Thursday 2-5:50** **Friday 12-3:50**	*(Homework Rec starts week #2)*

Text:	C++ Primer Plus, Stephen Prata Thinking in Java, Bruce Eckel
Canvas:	All required lecture videos, powerpoint slides, sample code, and lab materials may be found on Canvas under "Modules" Link: pdx.instructure.com
Disabilities:	If you have a disability and are in need of academic accommodations, please notify the instructor immediately to arrange needed support. This includes any accommodations required for taking examinations. All DRC exams must be taken at the same time as the class exams except when otherwise pre-authorized. Such pre-authorization should take place at least 24 hours prior to the class exam.
System & Compiler:	CS Linux (linux.cs.pdx.edu). C++ language implemented by the **g++** compiler. GNU GCC C++ compiler (g++) in the default –ansi mode Use the **C++ standard 98** guidelines (-ansi). **Editors MUST be either: vi, vim, emacs** ***No IDEs** are allowed when working in C++*

<u>Prior Knowledge expected:</u>
CS202 is designed for students who have already have programmed in C++ and are proficient applying recursion to data structure algorithms. Students should already be able to program each of the following using C++ at the linux command line, without an IDE:

(a) Classes, constructors, destructors
(b) Pointers and dynamic memory (new, delete)
(c) Pointer arithmetic
(d) Dynamically allocated arrays
(e) Linear, circular, and doubly linked lists
(f) Arrays of linked lists
(g) Binary search trees
(h) Be prepared to apply algorithms to implement balanced trees and graphs

Students should be able to demonstrate proficiency at applying recursive solutions to insert, traverse or remove from pointer based linked data structures of: linear linked lists, circular linked lists, doubly linked lists, arrays of linked lists, and binary search trees.

<u>Course Description and Goals:</u>
Students will become familiar with the language and operating system environment used in most upper division courses in the Computer Science major curriculum. Use of the file system, operating system calls, and shell-level programming; low-level debugging of high-level programs. Programming exercises will include applications of data structures and memory management techniques.

The primary goal in CS202 is to prepare students for programming in the upper division 3xx and 4xx level classes. To achieve this goal, CS202 focuses on three areas: object oriented programming, advanced C++, and an overview of how Java relates to what we have learned in C++.

The majority of the term will be spent introducing students to object-oriented programming while learning advanced C++ syntax. Students will understand the difference between procedural abstraction and object oriented solutions. Students will spend the term designing and programming with inheritance hierarchies, with the goal of solving problems efficiently: producing high quality, robust, maintainable as well as efficient object oriented solutions. This will provide students with the chance to experience object oriented design and programming. Programming assignments will focus on advanced data structures while at the same time accomplishing these other goals. Students will learn about C++'s function overloading, operator overloading, copy constructors, and be introduced to inheritance.

Java skills developed include writing two programs using advanced data structures in Java with strict requirements to follow OOP guidelines – all data members private, no friendly access, and complete implementation of functions required to handle issues of deep versus shallow copies and compares. Students learn the relationship between the two languages and the similarity of Java’s references to pointers.

Remember we are all in this together.

We all have real world struggles as we navigate this extraordinary time.

Remote for Fall 2020

This Fall, CS202 is offered as a remote class. We have adapted our material to be a mix of required content videos and live streaming sessions. The content videos must be watched prior to the live streaming sessions each week per our "Course Plan". The live streaming sessions will take place during our scheduled class period (T/TH 4:40pm) and typically end by 6:10pm. We use zoom for live streaming lectures and lab sessions.

Attendance to the live streaming sessions is part of your success in this class and is required; missing sessions means that success will be jeopardized. Material will be covered in the live streaming sessions that cannot be found in any other format. If you are unable to attend these sessions, contact karlaf@pdx.edu to arrange an alternative (such as through watching recordings of these sessions). Approval for such an arrangement will be made on a case by case basis and in such situations the student agrees to watch all recordings for the missed live streaming sessions. **It is important to note that the lab sessions are not recorded and attendance is required.**

You are Granting Your Permission to be Recorded: Students understand that live streaming sessions may be recorded and this is being done with your permission. Such recordings are limited for use for just students in this course and may not be distributed to others outside our class roster. *This means you may not distribute videos or recordings of this class to other people outside of our class*.

Online examinations (including proficiency demonstrations) will take place using a recorded proctoring service. Students are required to have a webcam (USB or internal), a microphone, and internet access for these events. It is recommended to preplan to ensure that the space is private or semi-private and that the recordings that will be made do not invade any third-party privacy rights. Special arrangements may be made by contacting karlaf@pdx.edu.

All students must review the syllabus and requirements including testing requirements. Enrollment in this course is an agreement to abide and accept all terms.

Communicating in a Remote Environment

One of the pieces we miss by being remote is building a community. We will meet every Tuesday and Thursday at 4:40pm for our live streaming sessions (except for Thanksgiving!!). These are critical in order to "get to know" each other. Please plan on having your webcam on whenever you find it comfortable to do so and participate with other students in the chat sessions. Of course, when using chat we are missing body language cues and immediate feedback; therefore, it is very important to be aware of online etiquette. This ensures that the message you intend to convey is received correctly.

When writing messages in the chat, keep in mind these points:

- **Be respectful.** It is essential to keep in mind the feelings and opinions of others, even if they differ from your own.
- **Discuss material related to the class.** Remember chat sessions are not the same as personal social media.
- **Be aware of strong language, all caps, and exclamation points.**
- **Be clear with humor and sarcasm.** Is what you are saying clear?

Proficiency Demonstrations:

- Twice a term we will be evaluating coding proficiency using live programming examinations called Proficiency Demonstrations, set up by appointment.
- Every student in CS202 must show proficiency in programming in C++, recursion and data structures, using linux with either vi, vim, or emacs
- Proficiency demos must be passed to receiving a passing grade in CS202
- Proficiency demos are scored as:
  - E (exceeds, passing),
  - P (proficient, passing),
  - PW (pass with warning, passing)
  - IP (in-progress, non-passing), ** maybe retaken once at midterm time
  - U (unsatisfactory, non-passing)
- A PW (pass with warning) at the final time is an indicator that students should continue to practice before taking upper division courses such as with CS299
- There are no re-tests available for the final proficiency demo.

Lab Sessions:

The lab sessions are where we reinforce the materials learned in lecture. It is where concepts will be practiced prior to applying them to the larger individual programming assignments. *It is expected that all students will attend the lab section enrolled in, each week.*

Labs begin week #1. Attendance is required. Starting the second week, completion of a prelab quiz on Canvas is required in order to attend lab. **These need to be done by 10am on Friday of each week (starting week #2).** Prelabs help ensure that students are ready for the lab experience; if you can’t complete the prelab then it means additional studying should be done prior to the lab so that you can make the most of your time with our technical assistants. Seek assistance through homework recitation, office hours or by appointment.

Some key points about labs include:

1. **Attendance** is required and labs are scored pass/no pass.
2. Refer to the Course Plan to learn which lab is being performed each week.
3. The lab materials can be found on Canvas in the Lab shell (CS202L) and it is organized by the Lab number.
4. **Read** the background for each lab, **prior** to attending the corresponding lab!
5. **REQUIRED: Perform the prelab quiz** on Canvas prior to attending the lab
6. **Plan** to fill out all of the questions in the lab materials as you progress through the lab.
7. **On your own, make sure** to perform and practice all of the vim exercises assigned

<u>Individual Programming Assignments:</u>
There are three components to your programming assignments this term. We have design writeups with UML diagrams, the programming assignment code, and an efficiency writeup which is turn in with the code.

1. **Design writeups and UML Diagrams**
   a. There are 3 written design write-ups that are turned in **prior** to the corresponding programs. Included with each design write-up must be UML diagram; examples can be found in the background information for lab #1.. The design write-ups must be a minimum of 600 words.
   b. **Late write-ups** will be accepted by the late date **with a 5% deduction.**

2. **Individual Coding:**
   a. There are 5 programming assignments
   b. All programs must be created individually and written on linux.cecs.pdx.edu
   c. **Every** program must be submitted.
   d. **Passing scores** are expected **on ALL programming assignment to pass CS202**
   e. Turning in work by the late deadline is a 5% deduction
   f. **Assignments** may **not** be turned in later than the late due date
   g. **Programs incorrectly** uploaded will receive a 5% penalty.

3. **Efficiency and GDB Writeups:**
   a. A minimum 400 word written document analyzing your design and discussing how effective the classes that you created were. Discuss the validity of your approach and analyze it in terms of object oriented programming. **Think in terms of analyzing your solution!** This means discussing the efficiency of the approach as well as the efficiency of the resulting code.
   b. A minimum 200 word written discussion of how debuggers (gdb, xxgdb, ddd, etc.) assisted in the development. This write up must describe experiences with the debugger, how it assisted code development, or how it could be used to enhance the programming experience.
   c. These are turned in <u>with</u> your program
   d. Please do not combine the writeups with your tarball or zip file. They should be uploaded separately to Canvas.

4. Follow the STYLE SHEET!
   a. Program style and comments is 20% of each program's grade.
   b. Each file must have your name and header comments describing the purpose of the code within the file.
   c. The file header comments should be a **paragraph**, at a minimum
   d. Each function must have header comments describing the purpose of the function and arguments. No exceptions!
   e. Make sure that each function that has a non-void return type returns a value

   through each possible path
   f. Always use the returned value when calling a function
   g. Avoid single character variable names, except for loop control variables and array indices
   h. Avoid while(1) or while(true)
   i. Avoid the use of break or return from within a loop
   j. Very few functions in CS163 should have void return types.

5. Each program will have **multiple files** (.h and .cpp files for C++) and have their name and header comments in each file. This does not apply when working in Java.
   a. Group related functions and classes within a single file.
   b. Never place function implementations in a .h file and never #include a .cpp file!
   c. Limit your **C++** submissions to **FIVE** .cpp files in total or seek authorization otherwise. Do not assume that you can turn in more than this.
   d. **Before using tar or zip**, make sure to backup your files.
   e. **Check the archive**, to make sure that they are built correct and that you have not destroyed the contents of the file(s).
   f. **Check** CANVAS afterwards to make sure your files are correct.
   g. **There is a 5% deduction** for incorrectly archiving your files for submission

6. **Partial credit will be given for incomplete programs.** This means that it is better to turn in something, even if it doesn't work. If you find you are continually having problems meeting the due dates, make an appointment or visit office hours.

7. **Every assignment (programs and write-ups) must be submitted** to pass CS202.
   a. All code and written material must be your own work and may not be copied from the web or other students. Be careful to not plagiarize. Doing so will result in a zero on an assignment and a failure in the class. Receiving "to much help" is not a valid reason to receive a passing score.
   b. Never let anyone write the code for you! Never just copy code from the internet!

Term Paper:

- Each student will be required to submit a typed term paper. The paper must **explore how well the programs written this term have met our objectives of being object oriented! Discuss how the designs met the criteria set out for OOP, and how they could have been improved.**
- The paper must be a minimum length of 4 pages and a maximum of 7 pages (double spaced, 12 point font) of text. This does not include the cover sheet, table of contents or reference pages.
- Attach any tables or sample code as exhibits; these may not be part of the 4-7 page count.
- The term paper should be done using **Word** or plain text. It may be submitted as a pdf if other word processors are used.

Overview of Grading Policies:

Demonstrate Proficiency in C++ - Midterm Demo - Final Demo	Pass/No Pass	Must pass both demos
Lab Participation - Lab Code Submitted	Pass/No Pass	Attendance is required
Individual Assignments - Term Paper - 3 Written Designs/UML - 5 Programming Projects	 5% 5% 20%	Submit to CANVAS Dropbox (All programs receive a passing score of 65% or greater)
Midterms - Midterm #1 - Midterm #2	 20% 20%	Exams must be 65% or greater to pass CS202
Comprehensive Final Exam	30%	***** Must receive a Passing score of 65% to pass the class *****

Grading Policies:

1. Two midterms, each worth 20% of your grade
2. The Final Exam is 40% of your grade

• Exams will all be closed book, closed notes.

- **If a DRC exam is being taken, please email your teacher with a reminder that an exam is needed at the testing center.** Do not expect an exam to automatically be sent without such email.
- For C or better in this class, you must receiving a PASS on all of the Pass/No pass components of this class (see the chart above)
- **Both midterms and final** must receive passing scores to pass the class.
- Failure to turn assignments on-time or within the allowed late period will result in a zero for that assignment. Discuss extraordinary situations with your teacher.
- **GRADING** will be done near 90% (A-, A), 80% (B-, B, B+), 65% (C). Exact break points for grades will depend upon the overall class results. A **No Pass** on the proficiency demos or a failure to turn in an assignment will result in a non-passing grade (F, D-, D, D+).

- **No Basis for a Grade** applies when a student has not turned in any work and have not taken any exams. If you have complications and cannot finish the class, make sure to drop or withdraw. *Otherwise you will get a grade in the class.*

- **INCOMPLETES** will be given only when a **minimal amount of work** remains to be completed, only for a valid reason and only for a fixed time period. An Incomplete only applies in situations where the work already submitted has passing scores. ***Do not expect an incomplete in this class.***

CHEATING:

Each student is expected to submit only original work. **Any person who violates these requirements will receive a grade of zero for an assignment which based on the above grade requirements will result in an F for the course.** A letter will be sent to the head of the CS Department.

The work you submit must be your own. It is not acceptable to hand in assignments in which substantial amounts of the material was done by someone else. You must be especially careful that in the process of discussing problems with other students that they do not inadvertently end up using your work. In such an event, all students involved will receive a zero on that assignment.

Students will receive a zero on an assignment if any of these activities take place:

1. Student provides proficiency demo questions to other students
2. Student provides proficiency demo solutions to other students
3. Student solicits (asks for) proficiency demo questions and/or solutions from other students
4. Student copies lab code from another student
5. Student accepts an assignment and/or program from another student
6. Student supplies an assignment and/or program to another student
7. Student posts the assignment and/or program on the web, social networking site, or Canvas or Slack discussions
8. Student shares their password with another student at PSU giving that student access to their assignments and/or programs
9. Students work together on assignments or turn in the similar assignments.
10. Student turns in work that was obtained from other sources such as the web, friends, tutors or technical assistants
11. Student leaves work available for others to copy from
12. Student attempts to purchase programs from others (in person or electronically).

Seeking Assistance

Be careful when seeking help from others. We recommend seeking help from (a) instructor, (b) TA's and lab assistants, and (c) Tutors. Use caution otherwise. Do not to share your code with others! Never accept code that was not written by someone else! **This means, NEVER accept code from someone else, even if it is a tutor!**

- Never post your code in the Canvas or Slack discussions, the web, or social networking sites.
- Never give your assignments to other PSU students, regardless of their situation.
- Never email your code to anyone except your instructor.
- I recommend NEVER having someone else use your keyboard or type code for you when asking for help. Ask them to teach you what you need to know rather than do it for you!!!

Let's have a GREAT FALL!