**CSS 143 B Au 19: Computer Programming II**

[Jump to Today](https://canvas.uw.edu/courses/1331881)

**Vitals**

* Course Number and Title: CSS 143 (Section B): Computer Programming II.
* Description: "Transition from basic programming skills to a rigorous process of software development. Familiarization with higher level programming techniques (recursion, generic programming, stacks, queues, trees, searching, and sorting). Emphasizes connection between algorithmic thought and implementation. Prerequisite: minimum grade of 2.7 in CSS 142; co-requisite: CSSSKL 143." ([University of Washington Bothell 2017 (Links to an external site.)](http://www.washington.edu/students/crscatb/css.html))
* Meeting Time/Place: TTh 5:45 pm–7:45 pm (UW1-202).
* Credit: 5 units. This translates into 10–15 hours of work per week *outside* of lecture.
* Contents of this Syllabus and the course web site can change at any time in the term.
* Class Culture:
  + Excellence: I have high standards. You are expected to meet them.
  + Initiative: Ask questions! Engage and wrestle with the material, your colleagues, and myself.
  + Grace: Freedom to learn and grow. Extend courtesy to each other.

**Instructor**

* Name: Johnny Lin.
* Email: jwblin@uw.edu.
  + Please assume I check email only 1–2 times/day and email me only for simple questions.  I also **do not check email much on Sundays.**
  + I have lots of [office hours (Links to an external site.)](http://www.johnny-lin.com/sched/) so you can come ask me more complex questions (or to talk about anything)!  You don't need some serious reason ... feel free to just come by!
  + Remember that due to [FERPA rules (Links to an external site.)](http://www.washington.edu/students/reg/ferpafac.html), I cannot email you grades.
* Twitter: jwblin (I don't post very often, however).
* Work Phone: 425-352-5374.
* Office Location: UW1-260R.
* Office Hours: See [my schedule (Links to an external site.)](http://www.johnny-lin.com/sched/) for my regular office hours, make an appointment to see me, or just drop in if my door's open.
* Personal Home Page: [http://www.johnny-lin.com (Links to an external site.)](http://www.johnny-lin.com).
* [Student Teaching Evaluations (Links to an external site.)](http://www.johnny-lin.com/teach/evals/)
* [My RateMyProfessors.com Rating (Links to an external site.)](http://www.ratemyprofessors.com/ShowRatings.jsp?tid=1983565)

You must also be registered for one of the accompanying labs to this course (CSSSKL 143), each of which is taught by another instructor. Please see your lab section's web page for details (it should appear automatically on your Canvas site).

Acknowledgements:  The components and structure of this course is a collaborative effort over many years and by many instructors in the CSS Division and some of that work is used verbatim in course documents without individual attribution or quotation.

**Texts & Supplies**

**Required text:** Savitch, Walter, *Absolute Java,* 6th Edition, Pearson, 2016, ISBN 0134041674. In documents in this course, you might see the text abbreviated "S".

**Additional resources:**  We may have other readings during the term; they will be available online via Canvas or distributed in hard copy.  Some/all of the lectures will be available as Panopto recordings.

Programming is learned primarily through practice, both writing and reading practice.  The examples in the textbook as well as the Self-Test Questions and Programming Projects in the textbook offer you copious practice opportunities.  Here are two more sites that give you exercises to do:

* [Codingbat (Links to an external site.)](http://codingbat.com/)
* [Practice-It (Links to an external site.)](http://practiceit.cs.washington.edu/)

There are, of course, many web resources and tutorials on Java. Feel free to use any that will help you get the practice you need! Just be careful not to choose ones that are too easy; some tutorials won't challenge you enough.

**Computer requirements:** All students are expected to have unfettered access to a personal computer (Windows, Mac, Linux, etc.) or netbook with broadband-speed Internet access. Student access to University computing resources fulfills this requirement. All students are assumed to be proficient with an office productivity suite and email. All students must [enable their Canvas Notifications for this course (Links to an external site.)](http://guides.instructure.com/m/4152/l/73162-how-do-i-set-my-notification-preferences) to be at least as frequent as the defaults; if I post announcements, etc. via Canvas, you need to be able to receive them. All students must have a valid email account that they check on a *daily* basis.

**What to bring:** To each lecture:

* A laptop or tablet: We'll do some coding on a computer in lecture, but mostly your device will be used for taking quizzes, exams, etc. If you don't have a personal device, please see me.
* The textbook: We'll be talking through examples, etc.
* A writing tool and paper: Most of the work we do with programming will be done by hand. This is because coding is **not** about typing things into a computer but about thinking how to tell the computer what to do.

**Goals**

From the *Guidelines for Teaching CSS 161 and 162* document (CSS 142 used to be called CSS 161 and CSS 143 used to be called CSS 162):

The CSS 162 learning outcomes are: "...to facilitate a transition from basic programming skills to a rigorous process of software development. You will become familiar with higher level programming techniques (object‐orientation, recursion, inheritance, generic programming), constructs (lists, stacks, queues, trees) and algorithms (searching, sorting). The analysis of algorithms is introduced."   
The higher‐level programming techniques are aligned with the idea of moving from solving small problems that almost directly translate into small code (in 161) to solving problems that require multi‐step approaches or sub‐problems that themselves must be decomposed into components and algorithms. Students get used to the idea that it isn't always the case that they can see a problem solution right off the bat: that they need to proceed a step at a time and that there are heuristic processes, program structures, and standard algorithmic approaches that can help them manage their software development tasks.

Last, but not least, I want you to grow in learning how to learn. All my pedagogy has this as a core outcome.

**Requirements**

**Reading Quizzes (RQ) and Reading Activities (RA):** Reading quizzes (when they occur) are the first 10 minutes of the lecture. The best way to study for these quizzes is to go through all the Self-Test Questions in the book (the solutions to those questions are at the end of each chapter). Questions on each Reading Quiz will be similar (or even identical) to the Self-Test Questions.  RAs are other activities (e.g., worksheets) designed to help you engage the reading material.  RQs are generally graded on right/wrong while RAs are generally graded on effort.

**Homework (HW):** More integrative projects that solve a problem/do a task. (As opposed to labs which tend to be more focused on practicing one element of programming.) Please remember that learning programming cannot be crammed in. (In contrast, once you have learned how to program, you can write a program in an all-night frenzy if you want.) You must *start the assignments early and see me promptly* if you get stuck. Otherwise, you will likely go around in circles wasting time and energy.

*Programs that compile and run:* Because everyone has different computers, it is difficult to make sure we can execute your code on our machines.  Thus, it is your responsibility to make sure the following two conditions are met for all programs you turn in:

* It runs in BlueJ (You may use any IDE you prefer, but it is not possible for us to support every possible IDE, and some IDEs add extra lines of code that make it impossible to run on another machine. So please check it runs in BlueJ before submitting your code.)
* It has no environment-specific settings (e.g., absolute paths), unless otherwise permitted in the assignment description

If your program does not meet the above two criteria and the grader cannot get it to run and/or if the grader needs to change your code to make it work, you will receive no credit in the pertinent criteria.

*Passing sample drivers:*  In many of the homework assignments, we provide a sample, bare-bones driver.  At the very minimum, your code has to pass that driver.  However, for most assignments, we ask you to create your own driver that is much more elaborate than the sample driver.  In those cases, your code should pass both the bare-bones driver we gave you as well as the more elaborate driver you wrote.

**Midterms (M):** Mixture of multiple-choice and handwritten questions.

**Final Exam:** Cumulative. Mixture of multiple-choice and handwritten questions.  "If a student has three exams on one day, he/she should contact their advisor during the first three weeks of the term to work with the faculty to reschedule one of the exams to a different day." (Final Exam Policy for the School of STEM 2014)

In general, you will take quizzes and exams online in Canvas and submit your work online via Canvas. All exams must be taken in the exam room unless you have made alternative arrangements with the instructor.

**Grading**

The UW grading system is described [in detail in the Student Guide (Links to an external site.)](http://www.washington.edu/students/gencat/front/Grading_Sys.html). All graded work in this class is scored on points.  All the points in a category (e.g., HW, RQ and RA, etc.) are added up and that total is converted to a percentage. (Note, see the [Assignments listing](https://canvas.uw.edu/courses/1331881/assignments) for any rules applied to a category, e.g., dropping the lowest grade in the category, [in terms of negative impact (Links to an external site.)](https://help.instructure.com/entries/21413518-How-do-I-drop-the-lowest-score-s-from-a-gradebook-category-).) The category weighting is [then applied to that *category* percentage (Links to an external site.)](http://sites.ewu.edu/elearningservices/set-up-weighted-grading/), which then becomes part of your final grade. Thus, small point value assignments will be worth less in terms of its category than large point value assignments.

Your final grade, as expressed in grade points, will be calculated from your percentage grade assuming 60% is equivalent to a grade of 0.7 and 97% is equivalent to a grade of 4.0. Grades higher than 97% receive a grade of 4.0 and grades lower than 60% receive a grade of 0.0. I plan to [linearly interpolate (Links to an external site.)](http://en.wikipedia.org/wiki/Linear_interpolation) percentage grades in between 60 and 97% into grade points, using these endpoints, using the [Canvas conversion utility (Links to an external site.)](http://www.washington.edu/itconnect/learn/tools/canvas/canvas-help-for-instructors/assignments-grading/create-grade-scale/) (which results in a stepped rather than continuous conversion table). Images of the conversion table are given in these files:  [grading\_scale\_1.png](https://canvas.uw.edu/courses/1331881/files/58017711/download?wrap=1), [grading\_scale\_2.png](https://canvas.uw.edu/courses/1331881/files/58017715/download?wrap=1), and [grading\_scale\_3.png](https://canvas.uw.edu/courses/1331881/files/58017723/download?wrap=1).

Note that the grading methodology may change slightly based upon the capabilities of Canvas and other similar limitations.

Details of the UWB grading system and policies is available [here (Links to an external site.)](https://www.uwb.edu/registration/policies/grading/grading-ug). Please pay particular attention to the policies regarding Incompletes.

**Make-Up and Late Policy**

Work is not accepted late, nor are there makeups for any work, unless for reasons of physical health, bereavement, University business, or similar. Please see me if something like this comes up.

Extensions must be requested in a timely manner.  For instance, unless you are physically unable to send an email, you should notify me reasonably close to the exam time if you're too ill to take the exam.  If you have University business that conflicts with an exam, you need to notify me immediately upon learning of the conflict.  Any non-timeliness of an extension request is a prima facie reason for me to reject the requested extension.

NB: In a CSS class, "my computer crashed before (or after) I saved" is not an acceptable excuse.  It's like telling your math instructor you couldn't do your homework because you couldn't find your calculator.  Save often!  Make backups!  The easiest way of doing the latter is to write all your code in a folder that is synced to [Dropbox (Links to an external site.)](http://www.dropbox.com).  If you don't know how to make backups, please see me.

**Collaboration**

As with any relationship, I expect honesty, respect, and grace to be the foundation of our time together. I expect this for myself, as well as for you. Collaborating in work is a wonderful opportunity to practice all three principles, and mastery of content in the course is only possible with the practice of these three virtues. Cheating, however, is an affront (to the first two) or an abuse (of the last) of these norms, and so will result (at a minimum) in a zero for the Assignment involved. If you are not sure what constitutes cheating, please ask me; here are some guidelines to help you on this issue:

* For problem sets, papers, reports, presentations, and projects: I encourage you to work together and discuss your ideas freely. Each person must, however, write-up the Assignment themselves; do not just re-copy one another's solutions.
* For writing assignments of any form, plagiarism is, of course, cheating. Plagiarism involves representing another's work as your own and/or without proper acknowledgment. The most common form is the use of another's words without quotation or citation. UWB Learning Technologies has created [this page (Links to an external site.)](http://www.bothell.washington.edu/learningtech/instructional-resources/plagiarism) that gives more information about plagiarism and how to prevent it.
* For the quizzes and exams: These should be your own work, written without the use of other help or resources, unless explicitly specified.

While I take steps to decrease temptations to cheat (e.g., limit accessibility of solutions), with the advent of the Internet, it is often not that difficult to obtain copies of exams, problem sets, and/or solutions. However, I ask that current students limit their consultation of work from previous sections of the course in certain ways. Here are specific scenario categories:

* Consultation of problem set solutions from previous sections *while you are working on that same problem*: This constitutes cheating.
* Consultation of papers, reports, projects or presentations from previous sections while working on a current section's paper, report, project or presentation: This does not constitute cheating. I encourage you to use other people's work to give you ideas (though, as mentioned above, please do not just copy their work) for these types of Assignments!
* Consultation of Assignments besides quizzes or exams (e.g., problem set solutions, papers, etc.) from previous sections *while studying* for a current section's quiz or exam: I do not consider this cheating.
* Consultation of quizzes or exams from previous sections *while studying* for a current section's quiz or exam: I *do* consider this cheating. Please do not engage in this activity. The only reliable way to prevent this activity is to always collect all quizzes and exams immediately after students have taken the quiz/exam. This prevents current students, however, from benefiting from their own quizzes in studying for the final. Thus, I generally allow students to keep their quizzes and request future students to refrain from consulting those past quizzes, unless I provide them.

For example, if you have a friend who took this course before and you consult his/her problem set solutions while studying for an exam, that does **not** constitute cheating. However, if you consult those problem set solutions while working on that problem set, that does constitute cheating.

I believe these are reasonable guidelines that balance a number of competing interests and that do not constitute an unfair burden on conscientious students. I do not curve grades, so dishonest students really are hurting themselves and their own mastery of the content, while not hurting other students. You should also familiarize yourself with the [Student Guide to Academic Integrity (Links to an external site.)](http://www.uwb.edu/studentservices/student-guide). Please let me know if you have any concerns.

**The Internet and Programming:** The Internet is one of the greatest programming helps ever invented. Almost any programming problem can be solved using Google. That being said, you still need to take care not to plagarize another's code. Here are some guidelines:

* If you use a few words of someone else's code: If it's obvious, you probably don't need to attribute this use. If it's not obvious, you should put a comment line in saying where you got it from (the URL, at least).
* If you use substantial amounts of someone else's code: You must put a comment line in saying where you got the block of code from.
* If you use ideas from someone else's code: Obvious ideas probably don't need to be attributed. Non-obvious ideas do, and you should put a comment line in saying where you got the ideas for your code from. This will help you track the context of your solution.
* If you copy code from a copyrighted source: Directly copying is probably fine for personal study, but it is **not fine** for any programs you later distribute. You could get into serious legal trouble if you use someone else's copyrighted code without their permission. This is true even if you attribute your source, or if you give away your program for free. If you do use copyrighted code, make sure you note in your program you have done so, so if you ever do distribute your program you'll know whether you're in potential legal jeopardy. Or better yet, just never use copyrighted code in your programs without their permission!

**Due process:** As a student in a State institution, it is your right to be afforded due process in accordance with the 14th Amendment to the U.S. Constitution. Accordingly, if you believe any grade-related sanction I propose is inappropriate, you have the right to appeal the sanction to the Dean's Representative. The *Faculty Resource on Grading's* discussion of [due process (Links to an external site.)](http://depts.washington.edu/grading/conduct/dueprocess.html) and [academic misconduct (Links to an external site.)](http://depts.washington.edu/grading/conduct/prevention.html) has additional information. The [Student Code of Conduct (Links to an external site.)](http://www.washington.edu/cssc/student-conduct-overview/student-code-of-conduct/), of course, is controlling.

**Access and Accomodations**

"Your experience in this class is important to me. If you have already established accommodations with Disability Resources for Students (DRS), please communicate your approved accommodations to me at your earliest convenience so we can discuss your needs in this course.

If you have not yet established services through DRS, but have a temporary health condition or permanent disability that requires accommodations (conditions include but not limited to; mental health, attention-related, learning, vision, hearing, physical or health impacts), you are welcome to contact DRS at 425-352-5307 or drs@uwb.edu. DRS offers resources and coordinates reasonable accommodations for students with disabilities and/or temporary health conditions. Reasonable accommodations are established through an interactive process between you, your instructor(s), and DRS. It is the policy and practice of the University of Washington to create inclusive and accessible learning environments consistent with federal and state law.

For our Veterans: If you are a student who has served in our nation's military forces, thank you for your service. I hope that you feel comfortable enough to confidentially self-identify yourself to me so I can help you make a successful transition from the military to higher education." (Disability Resources for Students Syllabus Statement)

"Washington state law requires that UW develop a policy for accommodation of student absences or significant hardship due to reasons of faith or conscience, or for organized religious activities. The UW's policy, including more information about how to request an accommodation, is available at [Faculty Syllabus Guidelines and Resources (Links to an external site.)](https://registrar.washington.edu/staffandfaculty/syllabi-guidelines/). Accommodations must be requested within the first two weeks of this course using the Religious Accommodations Request form available at: [https://registrar.washington.edu/students/religious-accommodations-request/ (Links to an external site.)](https://registrar.washington.edu/students/religious-accommodations-request/). " (Religious Accommodations Policy)

**Class Cancellation Policy**

If classes are cancelled because of weather or other situations that impact the entire campus, please see the [UW Bothell main page (Links to an external site.)](http://www.bothell.washington.edu/). You will probably want to [sign-up for UW Alert (Links to an external site.)](http://www.washington.edu/safety/alert/) to be notified of any emergency situations. The [campus emergency page (Links to an external site.)](http://www.bothell.washington.edu/emergency) also contains helpful information. For instructor-related cancellations (for instance, if I'm sick), I will post an announcement on Canvas.

**Schedule**

This schedule below is sorted forward-chronologically. The schedule is tentative and may change as the quarter progresses.

**Course Summary:**

| **Date** | **Details** |
| --- | --- |
| Thu Sep 26, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Preliminaries and Review of CSS 143](https://canvas.uw.edu/calendar?event_id=1281158&include_contexts=course_1331881) | 5:45pm to 7:45pm | |
| Tue Oct 1, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Introduction to Classes](https://canvas.uw.edu/calendar?event_id=1281144&include_contexts=course_1331881) | 5:45am to 7:45pm | | Assignment | [RA (Syllabus, Coding Style, S4.1-4.2)](https://canvas.uw.edu/courses/1331881/assignments/4923157) | due by 5:45pm | |
| Thu Oct 3, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Composition and Information Hiding](https://canvas.uw.edu/calendar?event_id=1281145&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S4.1-4.2)](https://canvas.uw.edu/courses/1331881/assignments/4923131) | due by 5:55pm | |
| Sat Oct 5, 2019 | |  |  |  | | --- | --- | --- | | Assignment | [Fractions V1](https://canvas.uw.edu/courses/1331881/assignments/4923145) | due by 11:59pm | |
| Tue Oct 8, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Overloading, Constructors, and Encapsulation](https://canvas.uw.edu/calendar?event_id=1281146&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S4.3-4.4, 5.1)](https://canvas.uw.edu/courses/1331881/assignments/4923132) | due by 5:55pm | |
| Thu Oct 10, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Abstract Data Types](https://canvas.uw.edu/calendar?event_id=1281147&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RA (Abstract Data Types and Data Structures)](https://canvas.uw.edu/courses/1331881/assignments/4923151) | due by 5:45pm | | Assignment | [Fractions V2](https://canvas.uw.edu/courses/1331881/assignments/4923146) | due by 11:59pm | |
| Tue Oct 15, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Data Structures with Arrays](https://canvas.uw.edu/calendar?event_id=1281148&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S6, 14.1)](https://canvas.uw.edu/courses/1331881/assignments/4923137) | due by 5:55pm | |
| Thu Oct 17, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Event Handling, Exceptions, References, Parameters, and Testing](https://canvas.uw.edu/calendar?event_id=1281149&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RA (S9.1-9.2, 5.1-5.3, 6.2, p. 848)](https://canvas.uw.edu/courses/1331881/assignments/4923155) | due by 5:45pm | | Assignment | [Data Structures: ArrayList-like Structures, Stacks, and Queues](https://canvas.uw.edu/courses/1331881/assignments/4923143) | due by 11:59pm | |
| Tue Oct 22, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: More on References and Memory](https://canvas.uw.edu/calendar?event_id=1281159&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (5.1-5.3)](https://canvas.uw.edu/courses/1331881/assignments/4923135) | due by 5:55pm | |
| Thu Oct 24, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Inheritance, Packages, and Documentation](https://canvas.uw.edu/calendar?event_id=1281150&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RA (S7, 5.4)](https://canvas.uw.edu/courses/1331881/assignments/4923154) | due by 5:45pm | | Assignment | [Classes and Privacy Leaks (Money, Date, Bill)](https://canvas.uw.edu/courses/1331881/assignments/4923142) | due by 11:59pm | |
| Tue Oct 29, 2019 | |  |  |  | | --- | --- | --- | | Assignment | [Answers to Questions 1](https://canvas.uw.edu/courses/1331881/assignments/4923138) | due by 5:45pm | | Assignment | [Midterm 1 (Part 1)](https://canvas.uw.edu/courses/1331881/assignments/4923126) | due by 7:45pm | | Assignment | [Midterm 1 (Part 2)](https://canvas.uw.edu/courses/1331881/assignments/4923148) | due by 7:45pm | |
| Thu Oct 31, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Recursion](https://canvas.uw.edu/calendar?event_id=1281152&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S11.1-11.3)](https://canvas.uw.edu/courses/1331881/assignments/4923136) | due by 5:55pm | | Assignment | [Shape Inheritance](https://canvas.uw.edu/courses/1331881/assignments/4923159) | due by 11:59pm | |
| Tue Nov 5, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Recursive Searching](https://canvas.uw.edu/calendar?event_id=1281153&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RA (S11.1-11.3)](https://canvas.uw.edu/courses/1331881/assignments/4923152) | due by 5:45pm | |
| Thu Nov 7, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Data Structures (Links and Nodes)](https://canvas.uw.edu/calendar?event_id=1281151&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RA (S13.2, 15.1-15.2, 15.4)](https://canvas.uw.edu/courses/1331881/assignments/4923153) | due by 5:45pm | | Assignment | [Recursion](https://canvas.uw.edu/courses/1331881/assignments/4923158) | due by 11:59pm | |
| Tue Nov 12, 2019 | |  |  |  | | --- | --- | --- | | Assignment | [Answers to Questions 2](https://canvas.uw.edu/courses/1331881/assignments/4923140) | due by 5:45pm | | Assignment | [Midterm 2 (Part 1)](https://canvas.uw.edu/courses/1331881/assignments/4923134) | due by 7:45pm | | Assignment | [Midterm 2 (Part 2)](https://canvas.uw.edu/courses/1331881/assignments/4923149) | due by 7:45pm | |
| Thu Nov 14, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Interfaces](https://canvas.uw.edu/calendar?event_id=1281154&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S13.1)](https://canvas.uw.edu/courses/1331881/assignments/4923133) | due by 5:55pm | | Assignment | [Linked Lists, Stacks, & Queues](https://canvas.uw.edu/courses/1331881/assignments/4923147) | due by 11:59pm | |
| Tue Nov 19, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Generics and Collections](https://canvas.uw.edu/calendar?event_id=1281160&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S14.2, 16.1-16.2)](https://canvas.uw.edu/courses/1331881/assignments/4923130) | due by 5:55pm | |
| Thu Nov 21, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Algorithm Analysis](https://canvas.uw.edu/calendar?event_id=1281161&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RA (S pp. 866-872)](https://canvas.uw.edu/courses/1331881/assignments/4923156) | due by 5:45pm | | Assignment | [Classes and Interfaces (Money, Etc. Revisited)](https://canvas.uw.edu/courses/1331881/assignments/4923141) | due by 11:59pm | |
| Tue Nov 26, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Sorting](https://canvas.uw.edu/calendar?event_id=1281155&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (S pp. 388-391, 701-708)](https://canvas.uw.edu/courses/1331881/assignments/4923129) | due by 5:55pm | |
| Thu Nov 28, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [No class: Thanksgiving Day](https://canvas.uw.edu/calendar?event_id=1281843&include_contexts=course_1331881) | 5:45pm to 7:45pm | |
| Tue Dec 3, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Data Structures (Trees)](https://canvas.uw.edu/calendar?event_id=1281156&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [RQ (15.7)](https://canvas.uw.edu/courses/1331881/assignments/4923127) | due by 5:55pm | |
| Thu Dec 5, 2019 | |  |  |  | | --- | --- | --- | | Calendar Event | [Lecture: Review](https://canvas.uw.edu/calendar?event_id=1281157&include_contexts=course_1331881) | 5:45pm to 7:45pm | | Assignment | [Pizza Simulator](https://canvas.uw.edu/courses/1331881/assignments/4923150) | due by 11:59pm | |
| Thu Dec 12, 2019 | |  |  |  | | --- | --- | --- | | Assignment | [Answers to Questions 3](https://canvas.uw.edu/courses/1331881/assignments/4923139) | due by 5:45pm | | Assignment | [Final Exam (Part 1)](https://canvas.uw.edu/courses/1331881/assignments/4923128) | due by 7:45pm | | Assignment | [F](https://canvas.uw.edu/courses/1331881/assignments/4923144) |  | |
| |  |  | | --- | --- | | [inal Exam (Part 2)](https://canvas.uw.edu/courses/1331881/assignments/4923144) | due by 7:45pm | |