JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Syllabus
Computer Science EN.601.220
Intermediate Programming - Online
Summer, 2021(4 credits, E)

*(*The instructors reserve the right to make adjustments to this syllabus as deemed necessary with notice.)

Course Description

- This course teaches intermediate to advanced programming, using C and C++. (Prior knowledge of these languages is not expected, but prior programming experience and a general understanding of basic object-oriented programming are expected.) We will cover low-level programming techniques, as well as object-oriented class design, and the use of class libraries. Specific topics include pointers, dynamic memory allocation, polymorphism, overloading, inheritance, templates, collections, exceptions, and others as time permits. Students are expected to learn the syntax and some language-specific features independently. Course work involves significant programming projects in both languages.
- **Prerequisites**
  Gateway Computing (AP CS, EN.500.112/113/114/132/133/134, or equivalent)
  Students are expected to be able to design and implement programs in a general-purpose programming language (such as Java or Python) prior to enrolling in this course.
- **Required** for CS majors and minors

Instructors Sing Chun Lee, Ph.D. Candidate
singchun.lee@jhu.edu, cs.jhu.edu/~singchun,
Office hours: to be announced (check the course Website)

Prof. Joanne Selinski, Associate Teaching Professor
joanne@cs.jhu.edu, cs.jhu.edu/~joanne/,
Office hours: see her website https://cs.jhu.edu/~joanne

Meetings

Venue: Online meetings on Zoom.
Zoom ID: 994 9122 6541
Passcode: see Blackboard
Time: Monday, Wednesday, Friday, 10:00am–12:15pm

Head Course Assistants

Srishti Dhamija (srishti@jhu.edu)
Office hours: to be announced (check the course Website)

Textbooks

- Recommended: Brian W. Kernighan & Dennis M. Ritchie, The C Programming Language, Prentice Hall, Inc., 2nd edition, 1988.
- Recommended: Andrew Koenig & Barbara E. Moo, Accelerated C++, Addison-Wesley, 2000.
- For a more detail-oriented presentation of C++, we recommend Deitel & Deitel, C++ How to Program; an electronic edition is available through MSEL: `Link`.
- Additionally, you will be expected to read various materials posted on the course website.

Online Resources

The following sites will be used heavily during the course:

- The course website is `jhu-ip.github.io/cs220-summer2021/`.
- Piazza (`piazza.com/jhu/summer2021/en601220`) will serve as our primary communication channel. You should sign up for Piazza immediately.
- Gradescope (`gradescope.com`) will be used for assignment submission, grades and feedback; you will receive an email invitation to this course on Gradescope.
- Blackboard (`blackboard.jhu.edu`) will be used for video distribution.
- (`en.cppreference.com`) and (`www.cplusplus.com`) provide excellent on-line language reference material.

Course Objectives

Upon successful completion of this course, you should be able to:

- Read, write, trace, test, and debug C and C++ program codes.
- Use command-line Unix/Linux tools for file management, creating and debugging programs in C/C++.
- Analyze a programming specification, design a program that fulfills the specification, create a development plan, execute the plan, implement and deliver the program on time.
- Utilize a distributed version control system to manage the development of a program.
- Understand the importance of good programming practices such as modularity, testing, documentation, and incremental design.

This course will address the following ABET Outcomes:

(1) (SO1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.
(2) (SO2) Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. in computing practice based on legal and ethical principles.
(3) (SO5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.
(4) (SO6) Apply computer science theory and software development fundamentals to produce computing-based solutions.

Course Topics

- General skills: code reading, tracing, and writing; pair programming and collaborative development; problem analysis and decomposition; incremental design; modular design; automatic testing
- Unix-like systems and tools: Unix basics, shell basics (command line tools, I/O redirection, etc.), shellscript and automation, text editors, gcc, make, gdb, valgrind, version control (git)

- C language, syntax, semantics, and concepts: types, operators, control structures, standard (terminal) I/O, scope, arrays, strings, pointers, dynamic memory allocation, structs, file I/O
- C++ language, syntax, semantics, and concepts: Standard Template Library (STL), containers (vs arrays), classes, templates, memory management, operator overloading, inheritance and polymorphism, object oriented design, exception handling

General Course Philosophy

This course will focus more on learning than on assessment. While we will use grades to let you know how you are doing, we hope that your goal is learning the material, rather than "getting a good grade." In the end, the best way to get a good grade is to develop an interest in learning and engage with the material in a self-directed fashion. Besides, in the long run, the knowledge and skills you acquire are far more important than the grade is.

That said, be aware that the main difficulty many students have with this course is **time management**. It will be a lot of work, and if you don't budget your time well, you may find yourself with a grade that does not reflect how well you understand the material. Additionally, while there are lots of resources provided to help you succeed, we cannot force you to use them; it is important to avail yourself of these resources, particularly office hours.

Course Approach We take a flipped-classroom approach to the course, with an emphasis on active learning during class sessions. Part of your homework will be to prepare for class by watching video lessons. During class sessions you will extend your learning by working with your classmates to discuss new concepts and apply them by completing in-class exercises, rather than listening to a lecture.

Because the course will be entirely online this semester with students in different timezones, we recognize that the ability to participate in class sessions might vary. We want to make sure that everyone is progressing satisfactorily. As such, students will be required to complete at least one individual code review with their instructor during the semester.

Expectations & Grading

All students are generally expected to attend all the online sessions of this course and actively participate by answering and asking questions and engaging with material via the posted material, exercises, assignments, and projects. Our online meeting time may involve any combination of recap of the material, discussion on the important concepts, review of previous exercise answers, as well as live coding exercises. Before each online session, students are expected to go through the related material posted on the course website and/or Piazza, and finish the coding exercises from the prior session.

Students will be expected to complete a variety of individual computer programming assignments, as well as two team projects. See the specific assignment page for details of what is permitted for a particular assignment. Failure to follow these guidelines will be a violation of the academic ethics code.

While code reuse is an important feature of modern programming, for this course, you will be expected to write most of the code for your homework assignments from scratch. You may use language libraries (according to assignment specifications), and you may always reuse your own code from prior work in the course. Downloading full or partial solutions from the internet, however, is an ethics violation. Using code from class examples, slides, or the textbook is acceptable, but you must cite the source properly in a comment in your code describing the original source.

There will be one midterm exam and a final exam. The midterm exam will cover the topics of the first half of the class (C material) and the final exam will focus on the second half (C++ material). The exams are designed to assess your knowledge of programming in C and C++ as well as your problem solving ability.

Homework assignments are expected to take a considerable amount of time; start early and budget your time well. On average, it should take about 20 hours per week. Additionally, try to use incremental development so that even if you run out of time, you can still turn in code that implements some of the desired functionality (with a README file and comments explaining what's missing). Keep in mind that half the features working all the way will get you a lot more partial credit than all the features half-way working. Good use of the version control system will significantly help with your incremental development. For each commit, you should update your README file to outline what's missing and what's done, and try to keep each commit a submittable version. This is a good practice, and can come in very handy in case you accidentally delete the local copy of your homework!

We will generally provide links to tutorials, references, and other resources for each topic. Students are expected to read these, as well as seek out other resources on their own to further their understanding of topics. There is a wealth of programming information on the internet; if one explanation doesn't make sense, you can probably find another that does.

Homework policy

Assignments will be due **by 11:00 pm on the due date** (unless otherwise indicated on the course website). **Non-compiling code will earn a score of zero**, so students are strongly encouraged to double-check that all submitted codes fully compile with no errors or warnings in the standard course compilation environment. We will accommodate late submissions in several specific ways:

- There will be a 45-minute grace period after the deadline during which assignments can still be submitted, but will receive a penalty deduction of 5% of the points possible on the assignment.
- Each student will be permitted to use up to 4 "late days" total during the semester on the individual coding assignments only. However, at most 2 late days may be used on any one assignment.
- Each delay of up to 24 hours past the 11:00 pm deadline for an individual coding assignment counts as one late day.
- No late days are allowed for partner-based coding projects.
- Students may not use a grace period with a late day. For example, if Student A turns in her homework at 11:50 pm on the day it is due (50 minutes past the original deadline), then she will be using one late day. Furthermore, if Student B turns in his homework at 11:15 pm on the day after it is due, he'll be using two late days.

Given these policies, please plan to get your homework done and turned in early so that if you encounter any last-minute delays, it will not hurt you too badly. Additionally, Gradescope will allow multiple submission attempts; by default, we will grade the last one. So it's a good idea to develop your program incrementally, and turn in a fully-compiling (if only partially complete) version every day or so.

Deadline exceptions can only be made by an instructor (not TAs/CAs), and will only be considered in the circumstances outside the control of the student (e.g., serious illness, death of a relative, etc.). If you must request an exception, do so as early as possible; it is easier to get an exception if you ask before an assignment is due, rather than after. No exceptions will be given for failure to plan ahead or simply having "too much work."

In-class Exercises

Many of the course topics will be supported by an exercise. These exercises are a very important part of your learning, and as such we expect you to complete them fully. Each exercise will specify what you need to submit in order to get credit for doing it. We will review the solutions to these exercises either fully or just the important parts in class sessions. We will *not* post solutions to the in-class exercises — if you are having trouble completing an exercise, seek help in office hours, or on Piazza.

Code Reviews

All students must satisfactorily complete at least one individual code review meeting with their instructor during the semester, within the requested timeframe. The purpose will be to review one or more homeworks that you've completed. These meetings will be about 20 minutes long and scheduled outside of class time, over zoom. At instructor discretion, some students might be required to meet for a 2nd code review to demonstrate their coding skills. **Although the code reviews are not graded, students who do not satisfactorily complete the required code review(s) will not pass the course.**

Grading Breakdown

- 12% - in-class exercises
- 28% - individual coding homework (4 total; due dates vary, will be listed on Course website)
- 14% - midterm coding project (in teams)
- 15% - midterm exam (Friday, July 02)
- 16% - final coding project (in teams)
- 15% - final exam (Friday, July 30)
- 0% - code review(s) - satisfactory completion required to pass the course

All scores and grader commentary on your homework and project submissions, as well as exams, will be available via Gradescope. Please keep your own record of your grades so that you will know your standing in the course. At the end of the term, letter grades are generally assigned according to the following scale. You should not expect a curve in this course.

$[97, 100]$: A+,	$[93, 97)$: A,	$[90, 93)$: A-
$[87, 90)$: B+,	$[83, 87)$: B,	$[80, 83)$: B-
$[77, 80)$: C+,	$[73, 77)$: C,	$[70, 73)$: C-
$[67, 70)$: D+,	$[60, 67)$: D,	$[0, 60)$: F

Classroom Climate

As your instructors, we are committed to creating an online classroom environment that values the diversity of experiences and perspectives that all students bring. Everyone here has the right to be treated with dignity and respect. We believe that fostering an inclusive climate is important because research and my experience show that students who interact with peers who are different from themselves learn new things and experience tangible educational outcomes. Please join us in creating a welcoming and vibrant classroom climate. Note that you should expect to be challenged intellectually by me, the course assistants (CAs), and your peers, and at times this may feel uncomfortable. Indeed, it can be helpful to be pushed sometimes in order to learn and grow. But at no time in this learning process should someone be singled out or treated unequally on the basis of any seen or unseen part of their identity.

If you ever have concerns in this course about harassment, discrimination, or any unequal treatment, or if you seek accommodations or resources, we invite you to share directly with the instructor or CAs. We promise that we will take your communication seriously and to seek mutually acceptable resolutions and accommodations. Reporting will never impact your course grade. You may also share concerns with the Department Head (Randal Burns, `randal@cs.jhu.edu`), the Director of Undergraduate Studies (Joanne Selinski, `joanne@cs.jhu.edu`), the Assistant Dean for Diversity and Inclusion (Darlene Saporu, `dsaporu@jhu.edu`), or the Office of Institutional Equity (`oie@jhu.edu`). In handling reports, people will protect your privacy as much as possible, but faculty and staff are required to officially report information for some cases (e.g. sexual harassment).

We hope that all students will be able to actively participate in all course meetings. Out of respect for your fellow students, be sure to join the Zoom session on time each day, use your full name in your zoom account settings, and add your picture to your Zoom profile. As instructors, we rely on non-verbal cues from your faces to guide us in class, so please do turn on your webcam as much as possible. Consider using a virtual background on days when you need some privacy. Please do unmute when collaborating with others, but be aware of potential background noises that could be distracting. If you miss a classroom meeting for whatever reason, you are responsible for the material presented.

Personal Wellbeing

- If you are sick, please notify us by email so that we can make appropriate accommodations should this affect your ability to attend class, complete assignments, or participate in assessments. The `https://studentaffairs.jhu.edu/student-health/`Student Health and Wellness Center is open and operational for primary care needs. If you would like to speak with a medical provider, please call 410-516-8270, and staff will determine an appropriate course of action based on your geographic location, presenting symptoms, and insurance needs. Telemedicine visits are available only to people currently in Maryland. See also https://studentaffairs.jhu.edu/student-life/student-outreach-support/absences-from-class/illness-note-policy/
- The Johns Hopkins COVID-19 Call Center (JHCCC), which can be reached at 833-546-7546 seven days a week from 7 a.m. to 7 p.m., supports all JHU students, faculty, and staff experiencing COVID-19 symptoms. Primarily intended for those currently within driving distance of Baltimore, the JHCCC will evaluate your symptoms, order testing if needed, and conduct contact investigation for those affiliates who test positive. More information on the JHCCC and testing is on the `https://covidinfo.jhu.edu/health-safety/johns-hopkins-covid-19-call-center/`coronavirus information website.
- All students with disabilities who require accommodations for this course should contact me at their earliest convenience to discuss their specific needs. If you have a documented disability, you must be registered with the JHU Office for Student Disability Services (Shaffer 101; 410-516-4720; `http://web.jhu.edu/disabilities/`) to receive accommodations.
- Students who are struggling with anxiety, stress, depression or other mental health related concerns, please consider connecting with resources through the JHU Counseling Center. The Counseling Center will be providing services remotely to protect the health of students, staff, and communities. Please reach out to get connected and learn about service options based on where you are living this fall at 410-516-8278 and online at `http://studentaffairs.jhu.edu/counselingcenter/`.
- Student Outreach & Support will be fully operational (virtually) to help support students. Students can self-refer or refer a friend who may need extra support or help getting connected to resources. To connect with SOS, please email deanofstudents@jhu.edu, call 410-516-7857, or students can schedule to meet with a Case Manager by visiting the Student Outreach & Support website and follow “Schedule an Appointment”.

Family Accommodations Policy

You are welcome to bring a family member to class on occasional days when your responsibilities require it (for example, if emergency childcare is unavailable, or for health needs of a relative). In fact, you may see my children in class on days when their school is closed. Please be sensitive to the classroom environment, and if your family member becomes uncomfortably disruptive, you may leave the classroom and return as needed.

University Policy on Incompletes

The university recognizes that the Summer 2021 semester is surrounded with uncertainty and many students may find themselves in unexpected situations where study is difficult if not impossible. Students who are confronted with extraordinary circumstances that interfere with their ability perform their academic work may request an incomplete grade from the instructor. While approval of such a request is not automatic, it is expected that faculty will make every effort to accommodate students dealing with illness in the family and other pandemic-related hardships. The instructor and student must establish a timetable for submitting the unfinished work with a final deadline no later than the end of the third week of the Fall 2021 semester. Exceptions to this deadline require a petition from the instructor to the student's academic advising office before that date. When entering an Incomplete grade in SIS, faculty must include a reversion grade which represents the grade the student will receive if s/he does not complete the missing work by the agreed-upon deadline.

For information on academic policies, see `https://e-catalogue.jhu.edu/engineering/full-time-residential-programs/undergraduate-policies/academic-policies/`

Ethics

The strength of the university depends on academic and personal integrity. In this course, you must be honest and truthful, abiding by the *Computer Science Academic Integrity Policy*:

> Cheating is wrong. Cheating hurts our community by undermining academic integrity, creating mistrust, and fostering unfair competition. The university will punish cheaters with failure on an assignment, failure in a course, permanent transcript notation, suspension, and/or expulsion. Offenses may be reported to medical, law or other professional or graduate schools when a cheater applies.
>
> Violations can include cheating on exams, plagiarism, reuse of assignments without permission, improper use of the Internet and electronic devices, unauthorized collaboration, alteration of graded assignments, forgery and falsification, lying, facilitating academic dishonesty, and unfair competition. Ignorance of these rules is not an excuse.
>
> Academic honesty is required in all work you submit to be graded. Except where the instructor specifies group work, you must solve all homework and programming assignments without the help of others. For example, you must not look at anyone else's solutions (including program code) to your homework problems. However, you may discuss assignment specifications (not solutions) with others to be sure you understand what is required by the assignment.
>
> If your instructor permits using fragments of source code from outside sources, such as your textbook or on-line resources, you must properly cite the source. Not citing it constitutes plagiarism. Similarly, your group projects must list everyone who participated.
>
> Falsifying program output or results is prohibited.
>
> Your instructor is free to override parts of this policy for particular assignments. To protect yourself: (1) Ask the instructor if you are not sure what is permissible. (2) Seek help from the instructor, TA or CAs, as you are always encouraged to do, rather than from other students. (3) Cite any questionable sources of help you may have received.
>
> On every exam, you will sign the following pledge: "I agree to complete this exam without unauthorized assistance from any person, materials or device. [Signed and dated]". Your course instructors will let you know where to find copies of old exams, if they are available.

In addition, the specific ethics guidelines for this course are as follows:

(1) This course is about learning, and encourages collaboration toward that end. In general, if a collaborative act helps you to learn, it is probably permitted. If, on the other hand, it helps you avoid learning, it is not permitted. For example, helping your friend learn how to use the debugger is great. Helping your friend by debugging their code for them is bad, because your friend will never learn how to do it by watching you. A main focus of this course is learning skills, and you can't acquire

skills without practice. Therefore, ”helping” other students by allowing them to skip the practice endangers the learning outcomes of the course, and is prohibited. Helping other students practice more efficiently and effectively (e.g. not waste 3 hours trying to fix one bug), on the other hand, actively supports the learning goals of the course, and is not only allowed, but encouraged.

(2) In general, when helping another student, never do something for them; instead, try to think like a teacher and ”teach” them how to do it themselves. This will help you both learn, since teaching something is a great way of learning more about it as well.
    - Asking a friend to let you look at their working code is not allowed, nor is offering to let someone else look at your working code.
    - If a friend is helping you to debug, you should only share the minimal amount of buggy code necessary.
    - In general, when helping others, think ”teach, not ”do”.
    - Always thank anyone who helped you on a given assignment in your README file.
    - The two coding projects for this course will allow for and encourage collaboration in teams. There are no limits to communication within a team, but work should be done ”as a team” with all members present; don’t just split the assignment into pieces and work on separate parts. No working source code should be shared outside your team.

Report any violations you witness to the instructor.

You can find more information about university misconduct policies on the web at these sites:

- For undergraduates:
  `https://studentaffairs.jhu.edu/policies-guidelines/undergrad-ethics/`
- For graduate students:
  `http://e-catalog.jhu.edu/grad-students/graduate-specific-policies/`