Course Syllabus

Overview

Welcome to Foothill College CS 2B, an intermediate computer programming course in C++. In this course, you will enhance your knowledge and skills in object-oriented programming. The most major topics of this course are inheritance and dynamic memory allocation, which will lead us to build more complex classes and ultimately some data structures.

Roadmap

The prerequisite for this course is CS 2A, where you learned some fundamental topics such as control flow, loops, arrays, and an introduction to object-oriented programming. In this course, you will refine your object-oriented programming skills, learn about memory management, and combine these topics and several others as you start to build data structures such as linked lists. At the end of this course, you'll be prepared for 2C, where you'll build more data structures and learn about tradeoffs when choosing the most advantageous data structure for the problem at hand.

Background, preparation, and growth!

Please be sure to review functions, loops, and arrays prior to the course, and come meet with me or with a tutor at the STEM center, or write in to our discussion board, if you have any questions. We will take some time at the beginning of the course to discuss object-oriented programming from 2A before going much more in depth with further topics.

The intermediate class is a HUGE time of growth. I have seen students come in who are very solid on the 2A material as well as those who have forgotten some material or feel a bit shaky on core concepts. No matter where you are coming in, you are welcome in this class, and you will learn a lot if you work hard. Please get in touch with me the first time you feel stuck in this class, even if you don't quite know what question to ask. I have a good track record of guiding students of all initial levels through their computer science courses.

Course logistics

Class meeting

Although this course is online, we have an optional in-person lecture on Mondays and Wednesdays, 4:30-6 pm at room 4218 (STEM center room next to CS lab). You are highly encouraged to attend the lectures to get the most out of this course.

This course is online. **However**, you must participate in the first two weeks of the course to avoid being dropped from the course (per California law, not modifiable by your professor!). Our "attendance" procedure (to show that you "attended" this online course) will be to introduce yourself in the discussion board, fill in the acknowledgement "quiz" which says you agree to the syllabus, briefly meet via video chat or in person with the professor, and turn in something of academic nature.

General time to set aside for the course

This is a core course of the CS sequence covering key concepts that will be important to your career in CS. The material covered in this course will require a substantial amount of time. Make sure to set aside at least 15 hours per week to cover the reading, practice questions, and programming assignments. I know it feels like a lot, but you will emerge super well prepared for CS 2C and/or your C++ career goals!

Important dates

Our final is a flexible-time, online final scheduled for March 26 noon - March 28 noon; you may choose any continuous window in this time to take the exam. Midterms will similarly be scheduled for February 14-16 and March 7-9. (Amount of allowed time TBD, but plan for a maximum of 2 hours for each of these exams).

Date range when you may take the exam	**Exam**
February 14 noon - February 16 noon	Midterm 1: Block off a continuous time block in this range to take your exam (1-2 hour, TBD)
March 7 noon - March 9 noon	Midterm 2: Block off a continuous time block in this range to take your exam (1-2 hour, TBD)

March 26 noon - March 28 noon	Final: Block off a continuous time block in this range to take your exam (1-2 hour, TBD)

The course schedule is listed at the bottom of the syllabus.

Text and references

The most important reference for the course will be the lecture slides which I've written and posted in the "Files" section. Please read these over and post any questions in the discussion board, or attend the in-person lectures and ask questions there.

The textbook for our course is Absolute C++ by Walter Savitch. The reason this book was chosen was partly so that those of you who took 2A last term and had this book could reuse the book. I have the 6th edition, but I read that the 3rd edition and onward should be ok for this course. I personally really liked this book. The book is a reference only; we will not have problems assigned out of the book. If you have any questions before purchasing the textbook, please let me know.

I have also posted modules that were authored by Professor Michael Loceff, another faculty member here at Foothill College.

We won't go in exactly the order of the textbook OR the modules; instead, take a look at the schedule at the bottom of the syllabus to find the relevant reference material for each week.

Instructor

My name is Professor Joanna or Professor Lankester; she/her/hers. I have worked as a software engineer and data scientist in multi-national corporations, startups, and my own small company, in addition to teaching. I love all things tech, have multiple projects going (always!), and write code every day. I also love teaching, mentorship, and Foothill College in general!

Office Hours

- Mondays after our lecture, 6-7 pm at the CS lab (STEM center room 4204) and by video chat
- Or by appointment (in person or on video chat) if that time doesn't work for you

If you need a 1:1 meeting (e.g. other topics that don't pertain to other students), you can either email me to set up an appointment, or wait around until others leave -- whichever you prefer.

Communication

Please post questions about the assignments in the discussions. If you have a question that is specific to you (e.g. registration issues, etc), you can message me on Canvas or email me at lankesterjoanna@fhda.edu.

Posting in Canvas discussions

The discussion board provides a great way to get quick help from each other and from me when completing the programming labs. Do make use of it!

Please do not post homework code, whether a question or an answer, in Canvas. Learning to write code involves synthesizing information, trying out examples on your own, and figuring out why the code may not be working the first time. Important to that process is your (and your classmates') opportunity to figure out the code without seeing the answer ahead of time.

When asking a question in the discussion board, make the question as specific as possible. If it's an error in your code you haven't been able to solve, describe what you've tried. The more specific your question, the more likely someone will be able to help.

Feel free to answer other students' questions; this is an encouraged and positive way to interact with classmates. Do not post homework solution code, but you can refer each other to where in the modules or book you found a similar example, or paste in some code from lecture that helped.

Software needed

The free and recommended Integrated Development Environment (IDE) for this course is Visual Studio for Windows and Xcode for Mac; the modules include instructions for setup. Of course, if you prefer a different IDE and can set it up on your own, or prefer an online code editor, you are welcome to use that instead.

Assessments and grading

The following assessments will constitute the indicated percentage of your final grade in the course:

Category	**Percentage**
Programming assignments	68
Practice Questions (PQs)	4
Midterm 1	6.75
Midterm 2	6.75
Final exam	13.5
Participation in surveys, discussion board, lecture, video chat meetings	1

Importantly, note that the exams (midterms + final) are mandatory. You cannot miss an exam and pass the class, even if the rest of your percentage score is over a passing grade. A passing grade must be earned on the final in order to pass the course.

Programming assignments

We will have 9 programming assignments (labs) in this course (8 regular-length and 1 mini-length). Programming assignments will **usually** be due at 2:59 pm (14:59). The first three will be due on Tuesdays, but the dates will move slightly throughout the term to give you an extra day on the weeks when we have a midterm. The labs must be your own work. You may discuss them with other students, but write up your individual solution.

The labs will constitute the majority of your effort in this course. The effort you put in will pay off, just as it does in other skills you learn, such as sports, arts, or musical instruments. Please set aside enough time each week to complete these assignments. I encourage you to put in the effort to think through the problems on your own, and then ask questions in the discussion forums as needed.

Code submission instructions

Please comment out your main using line comments (select the whole main, and use command + / on a Mac or Ctrl + / on Windows; or, look in your editor menu for something like, "Toggle comment"). Also put your output inside a comment block. Do not edit the output; paste it exactly as produced. Please submit one text file, or else whatever the assignment has requested.

Code should be submitted according to the style guide.

Lab grade policies

You will learn the most if you can fully debug your code and get it to run. Therefore late work will be accepted (at a 10% penalty per day) for up to 3 days. Code that does not run will be more heavily penalized, so it is worthwhile to complete the assignment even if late. Code is considered one day late starting immediately after the submission time, 2 days late 24 hours thereafter, and 3 days late 24 hours after that.

You may resubmit your code as many times as you want until the due date. After that, whatever I download when I start grading is the code that will determine your grade. Please send all files in one submission batch. Otherwise, if you submit files sequentially, then when I download the work, it will only download one of them. I will not start grading before the due date, so if you submit your work early and realize you want to update it before it's due, feel free; in this case, please remember to submit all files again at once.

I will ensure that the assignments are graded in a consistent way between students.

Code will usually, but not always, be graded according to the following categories:

Category	**Percentage Weight**
Output correct	22%
Testing complete	10%
Classes/methods/variables up to spec	25%
Code incorporates major concepts	35%
Style and clarity	8%

Some labs don't easily break down into these categories; for example, labs 3, 8, 9, and 10 (and possibly more) will just be graded with a particular number of points per section.

Recommendation: For 2B assignments, I personally find it easiest to print out the assignments on paper and write all over that paper. It allows you to circle or underline different things that need to be turned in, mark what you have and haven't completed so far, etc. When students are struggling to get their thoughts organized on programming assignments, printing out the assignment on paper is one of my first suggestions.

Practice questions

Practice questions, due on average 2-3 times per week, give you a chance to try a piece of code or concept that relates to the prior lecture material, but that is much smaller in scope than the programming assignments. They are great practice for the exam and/or for clarifying some details on sections that could be tricky prior to the programming assignment. Also, they are one of the most popular components of my courses! Time permitting, we'll work on them together in lecture sessions.

You can submit the answers to the practice questions on Canvas, or have me check them off in person at the lecture. You may type the answer on Canvas, unless a picture is required; then upload a photo of the picture you drew. Solutions will be posted on Canvas, but **please check your own answers; I will only spot check the PQs, not thoroughly grade them**.

As we get to coding questions later in the course, you may want to practice handwriting the code. Why? You will activate a different part of your brain when you write the answer instead of typing it. You will not rely on the IDE, but instead on your own knowledge. You will get the opportunity to practice writing code, which is common practice in software interviews. This last one is important. Trust me, you don't want your first experience writing code by hand to be during your interview!

Unlike the labs, the PQs are completely open to collaboration, so feel free to work together or work on them with a tutor at the STEM center. You may even copy the answer from the book, if you find it there (although I encourage you to try to think of the answer first). The only requirement is that you actually put down the answer. Turn them in on time for full credit or late for half credit.

Midterms and final exam

Please plan ahead for the exams. The exams will be scheduled at given day(s) with flexible windows, but must be taken in one block. The exams cannot be missed; they are mandatory to pass the course. Furthermore, passing the final is required to pass the course.

I author my own exams, which is to your advantage; you will have already practiced questions that the exam author also wrote. **The best way to study for the exams is to make sure you completely understand the practice questions (PQs).**You may want to work together, in the STEM center or virtually, to do the practice problems again to prep for the exams.

Exams cannot be copied or retained by students. Please note that many other materials are provided as learning tools; exams are assessment tools. Do not screen print, print, or otherwise retain copies of any exams. However, if you would like to review your exam after scoring, please set up an appointment with me, and I'd be glad to go over your exam with you.

Exams may include an oral component that consists of a short video or in-person chat in which you will verbally explain your answers to one or more questions. More details will be sent out at a later date via an announcement.

Regrades

I spend quite a bit of time reading and testing your code and providing you a high-quality, detailed review. I also give consistent grades across similar programs. For these reasons, I don't reconsider the number of points awarded based on the error made. I don't accept regrade requests on the basis that your code works on your computer, but not on mine.

However, please *do* let me know by email if you ever believe that I haven't seen one of your files or part of your code, for example, or if there is any ambiguity in the feedback provided.

NOTE: the official policy of this course is that any regrades, if granted, are subject to a full regrade, which could result in a lower score. Please send requests by email within 72 hours of a graded item being returned to you.

Questions about feedback

You are always welcome and encouraged to ask questions about the feedback I have written for your code. Some of the best learning happens during these discussions!

Extraordinary circumstances

Any extraordinary circumstances that interfere with your ability to complete work for this course will require official documentation.

Academic integrity

Programming assignments should be 100% your own work. Although you are free to discuss concepts with classmates, you should write up your own solution. Do not show other classmates your code, and do not look at anyone else's code. Do not send your code to another student; if they submit the code, neither of you would get credit, so don't put yourself in this situation. Steer clear of websites where solutions are provided; posting on these sites, in addition to bypassing your learning process, is potentially a copyright violation and a crime. No one, including any tutor, should ever be typing into your code. Your code must have been 100% produced by you.

Do not discuss the exam(s) with anyone. Exams must be taken alone. Retaining a copy of the exam is a violation of our academic integrity policy.

Any violation of the academic integrity clause of our course may be penalized up to and including a zero on the assignment and a referral to the dean of students.

I reserve the right to ask you to explain your code.

I diligently screen for matching code between classmates. This should be a relief to most students: you are working hard to get the work done, and I want to make sure you are getting the credit you deserve. You may find this surprising, but even within completely correct code on the same assignment, you all have an individual coding style, similar to a writing style or speaking style.

On-campus resources

Disability-Related Accommodations

If needed, please contact the Disability Resource Center (Links to an external site.)Links to an external site. as early as possible by visiting the DRC in room 5400, emailing the DRC at adaptivelearningdrc@foothill.edu, or calling DRC at 650-949-7017 to make an appointment.

The STEM Center

The STEM center in room 4213 offers free help to all students in science/math/technology courses and even has a separate Computer Science lab in room 4204. Please see more information including open hours at the STEM center website (Links to an external site.)Links to an external site..
We also have online help available during most evenings of the week. You may need to download a Blackboard program, so allow some extra time for that. Just go to this link and follow the directions to get through the downloads: Online CS help (Links to an external site.)Links to an external site.

Although I regularly work in the STEM center, I will need to help students in the order they arrive (including students from other courses) when I am scheduled there.

Update: I am not scheduled on a regular basis in the STEM center this term due to budget cuts. However, I will be there for high-volume or substitute days as follows:

CS lab: Monday, January 14 from 5-9 pm

STEM center: Tuesdays March 5, 12, and 19 from noon-5 pm

Psychological Services and Personal Counseling

Many students find themselves in a difficult or stressful time at some point. We are fortunate to have a great team of counselors (Links to an external site.)Links to an external site. who can talk with you about whatever you're going through. Please don't hesitate to contact them.

Questions?

If you have any questions about this course, please don't hesitate to message me and ask. The intermediate course is a time of growth not only in knowledge, but in general debugging and coding skills. I'm looking forward to this class, and I hope you enjoy it.

Schedule content

Here is the schedule of content for the course. The lecture pace, number and content of assignments, and assignment due dates are all subject to changes throughout the course, as I tailor every course to the unique cohort of students to optimize your education. Changes are likely to be very minor (e.g., more time needed in lecture on one topic and less on another).

Week number	Week start date	Topics
1	1/21/19	OOP review; constructors; const and static; enums; interface vs implementation; arrays

2	1/28/19	arrays with classes example; initialization section of constructor; linear vs binary search; strings in C++; binary and hex; cellular automata
3	2/4/19	inheritance; constructor chaining; pass-by
4	2/11/19	overloading; exception handling; pointers; dynamically allocated arrays
5	2/18/19	deep memory methods
6	2/25/19	multidimensional arrays; polymorphism
7	3/4/19	linked lists
8	3/11/19	templates; standard template library
9	3/18/19	trees; tree traversal; file I/O; multiple inheritance
10	3/25/19	review and final

The automated "course summary" below shows all due dates.

Things I'm required to include in this syllabus

Student Learning Outcomes - A successful student will be able to use the C++ environment to define the basic abstract data types (stacks, queues, lists) and iterators of those types to effectively manipulate the data in his or her program.A successful student will be able to write and debug C++ programs which make use of inheritance, i.e., the "is a" relationship, common to all OOP languages. Specifically, the student will define base and derived classes and use common techniques such as method chaining in his or her programs.A successful student will be able to define and use C++ templates to make their data and algorithms work with a variety of data types.

Number of credit hours that are considered hybrid per week: 2

Hybrid hours are: lab

Student attendance during hybrid hours is mandatory (this just means that your programming assignments are a component of the course).

Activities students need to do for hybrid course: programming labs

Course Summary:

Date	Details
Thu Jan 24, 2019	Assignment 0: Cards (to get quick feedback) - Optional

Date	Details
	Practice question 1
	Practice question 2
	Practice question 3
Mon Jan 28, 2019	Fill this out to "Attend" the course
	Introduce yourself
	Schedule initial meeting with professor
Tue Jan 29, 2019	Assignment 1 - Game Basics: Cards and Hands
Thu Jan 31, 2019	Practice question 4
	Practice question 5
Tue Feb 5, 2019	Assignment 3 - Cellular Automata
Thu Feb 7, 2019	Practice question 6
	Practice question 7
Tue Feb 12, 2019	Assignment 4 - Transactions with inheritance
	Practice question 8
Thu Feb 14, 2019	Practice question 9
Sat Feb 16, 2019	Midterm 1
Tue Feb 19, 2019	Practice question 10
Wed Feb 20, 2019	Assignment 5 - Complex Arithmetic with Operators
Thu Feb 21, 2019	Practice question 11

Date	Details
Fri Feb 22, 2019	Practice question 12
Tue Feb 26, 2019	Practice question 13
Wed Feb 27, 2019	Assignment 6 - Deep Memory Seven Segment Displays
Thu Feb 28, 2019	Practice question 14
Tue Mar 5, 2019	Practice question 15
Wed Mar 6, 2019	Assignment 7 - Seven Segment Displays on Consoles
Sat Mar 9, 2019	Midterm 2
	Practice question 16
Tue Mar 12, 2019	Practice question 17
Thu Mar 14, 2019	Assignment 8 - Soft delete in a Linked List
Fri Mar 15, 2019	Practice question 18
Tue Mar 19, 2019	Assignment 9 - Managing a Sorted STL List
Thu Mar 21, 2019	Practice question 19
	Practice question 20
Tue Mar 26, 2019	Assignment 10: Trees and templates
Thu Mar 28, 2019	Final exam
Tue Apr 2, 2019	Fall CS 2C survey