Computer Science Supplementary Authorization - 2023

Syllabus for SCI 5200

Instructor and Contact Information

Instructor: Melissa Danforth (pronouns: she/her)
Office Hours: Available on the Instructor's Contact Information page
Email: mdanforth@csub.edu

Virtual Class Information

This class uses a mix of pre-recorded videos and resources for the modules and weekly Zoom sessions (aka, this is a hybrid asynchronous and synchronous virtual course). The modules and Zoom information are posted on Moodle.

This class is centered around hands-on projects and the Zoom sessions will focus heavily on the projects. The Zoom sessions meet on Mondays from 4:00-5:00pm Pacific Daylight Time from Monday March 20th to Monday May 15th. There is no Zoom meeting on Monday April 3rd, as that is Spring Break week at CSUB.

Attendance during Zoom sessions is optional, but strongly encouraged so you can ask questions. Webcams are not required during Zoom sessions. I have also configured Zoom to allow phone call-ins and to mask phone numbers for those who have to call in to attend.

Recordings from the Monday Zoom sessions will be transferred over to Knowmia for post-processing and closed captioning before being posted to Moodle. This means there is a processing delay for the automatic video transfer between Zoom and Knowmia, and then for the manual closed captioning generation on Knowmia. Please allow 2 business days for the videos to be processed and posted.

General closed-captioning note: The closed captioning is generated by Knowmia's automatic captioning algorithm. I have edited some of the closed-captioning for the module videos to remove errors and clarify wording, but I did not have time to edit all of them. Also, I will n have time to edit the generated captions for the Zoom recordings. Please bear that in mind when watching the videos, but feel free to let me know if there's something very glaringly wrong about the captioning that I should fix.

The video transferred to Knowmia records the active speaker and shared screen from Zoom. This means the recording will capture anythi that is said over a microphone (and your webcam if you have it on), but it will NOT have the public chat log or the Zoom names associate with your mic/webcam feed. If you do not wish to appear in the recording, you can either public chat or private chat questions to me for n to answer during class. I'll anonymously repeat any chat questions before answering them.

Catalog Description

SCI 5200 Computational Thinking (3)
A graduate level course offered in an online format that satisfies the Computational Thinking requirement for an Introductory Supplementary Authorization in Computer Science. Topics include solving problems and designing systems using fundamental computing concepts such as decomposition, data representation, generalization/abstraction and algorithms. Students will learn effective computatio thinking and programming techniques for the classroom through study of historically significant works, conducting literature search in methods/case studies, and critical discussion in seminar setting.

Prerequisites

Admission to the M.A. in Education - Curriculum and Instruction (C&I) program or to the Introductory Computer Science Supplementary Authorization (CSSA) program.

Units and Contact Time

3 units lecture. 8 week virtual format with a mix of pre-recorded videos and weekly Zoom sessions. Self-paced. Plan to spend about 6-8 hours per week on videos, reading assignments, quizzes, and course assignments to complete the course.

Type

Satisfies the Computational Thinking requirement of the California Introductory CSSA.

General Course Notes

This course is organized by weeks to give you a recommended pacing through the modules to complete the course within the 8 week format. This is just a recommended pacing, and you are free to go on to modules in future weeks. Modules are roughly categorized into "Foundational Knowledge" (Weeks 1 through 4), "Teaching and Lesson Plans" (Week 5 and 8), and "Block Programming" (Weeks 5 through 8).

Individual modules generally contain four parts: (1) the learning objectives for that module, (2) required reading assignments, (3) optional resources that go further in-depth on the topic, and (4) my slides and video about the module. Some modules are on the honor system, where you check-off Moodle's completion box to indicate that you have completed the module. Some modules have an assessment quiz and/or assignment to gauge your learning. Assessment quizzes allow unlimited attempts, and the highest score is retained in the gradebook.

Later modules are locked to make sure you learn prerequisite knowledge and skills before attempting them. As you complete modules an get at least a 60% grade on the assessment quizzes, additional modules will be unlocked.

Collaborative and exploratory learning is encouraged in this course, with the below Academic Integrity note about turning in one's own work in one's own words. For most of these assignments, I am looking for an expression of creativity and critical thinking, which is a very individualized process, rather than one "correct" answer. It is exceedingly rare in computer science and computational thinking for there to only be one answer. Usually there are many different answers with different strengths and weaknesses, all of which are "correct" (i.e. they a solve the problem at hand). Please keep that in mind while you work on the assignments and discuss them with others in the course.

Class Principles

The following principles will guide this course:

- Communication: I fully understand if something unexpected has come up that interferes with your course work. Please communicate w me as soon as possible though, so we can discuss your options for moving forward in the course. Likewise, if something comes up with my situation, I will use the Moodle Announcement forum and Discord channels to keep you informed. It's important to keep the lines o communication open.
- Respect: There are many situations where differing, but equally valid, opinions and/or solutions may exist. Respect the rights of others t form different conclusions than your own. This includes engaging in respectful and professional discourse in the discussion forums.
- Critical Thinking: Most of the assignments in this course will require applying critical thinking and analysis skills. My grading approach f those "thinking questions" is more about seeing your thought process than seeking "perfect" answers. There are many possible and equally valid solutions to most computing problems, so it is far more important to explain your thought process for choosing your solution, rather than coming up with a "perfect" solution.

Required Textbook(s)

- Foundational Knowledge section: "Computational Thinking: A Beginner's Guide to Problem Solving and Programming" by Karl Beecher. BCS Learning & Development, 2017. ISBN 978-1-78017-36-41.
- Block Programming section: "Computer Science in K-12: An A to Z Handbook on Teaching Programming" edited by Shuchi Grover. Edfinity, 2020. ISBN 978-1-7346627-0-2.

Recommended Textbook(s) and Other Supplementary Materials

Recommended academic papers on computational thinking:

- Wing, Jeannette M. "Computational thinking." Communications of the ACM 49.3 (2006): 33-35. ACM: https://dl.acm.org/doi/10.1145/1118178.1118215
- Lee, Irene, et al. "Computational thinking for youth in practice." ACM Inroads 2.1 (2011): 32-37. ACM: https://dl.acm.org/doi/10.1145/1929887.1929902
- Brennan, Karen, and Mitchel Resnick. "New frameworks for studying and assessing the development of computational thinking." Proceedings of the 2012 annual meeting of the American Educational Research Association (AERA), Vancouver, Canada. Vol. 1. 2012. MIT: https://www.media.mit.edu/publications/new-frameworks-for-studying-and-assessing-the-development-of-computational-thinking/
- Weintrop, David, et al. "Defining computational thinking for mathematics and science classrooms." Journal of Science Education and Technology 25.1 (2016): 127-147. SpringerLink: https://link.springer.com/article/10.1007/s10956-015-9581-5
- Lye, Sze Yee, and Joyce Hwee Ling Koh. "Review on teaching and learning of computational thinking through programming: What is ne for K-12?." Computers in Human Behavior 41 (2014): 51-61. Science Direct: https://www.sciencedirect.com/science/article/abs/pii/S0747563214004634

Resources that I have used to develop modules, but that you DO NOT need to have or purchase for this course (this information is provide in the spirit of appropriately citing my sources):

- "Computational Thinking {and Coding} for Every Student: The Teacher's Getting Started Guide" by Jane Krauss and Kiki Prottsman. Corwin, 2017. ISBN 978-1-506341286.
- "The Power of Computational Thinking: Games, Magic and Puzzles to Help You Become a Computational Thinker" by Paul Curzon and Peter W. McOwan. World Scientific, 2017. ISBN 978-1-786341846.

Course Modules

Week/Section	Zoom Session	Online Modules	Assignments
Week 1 Foundational Knowledge	Overview of the course and syllabus	Module 0 - Demystifying Computer Science Module 1 - History of Computer Science Module 2 - Overview of Computational Thinking	Complete Your Virtual Rolodex Ent National Geographic Citizen Scienc Course Computing Diversity Discussion Forum Checkpoint Assignment 1: Choose Project Topic
Week 2 Foundational Knowledge	Walking through Checkpoint Assignment 1	Module 3 - Algorithmic Thinking Module 4 - Decomposition	Quiz on Algorithmic Thinking Quiz on Decomposition Checkpoint Assignment 2: Decomposition of Project
Week 3 Foundational Knowledge	Walking through Checkpoint Assignment 2	Module 5 - Pattern Recognition and Generalization	Quiz on Pattern Recognition and Generalization Checkpoint Assignment 3: Pattern Recognition and Generalization
Week 4 Foundational Knowledge	Walking through Checkpoint Assignments 3 and 4	Module 6 - Abstraction and Modeling Module 7 - Evaluation Module 8 - Cybersecurity Concepts for Algorithms and Program Design (optional)	Quiz on Abstraction and Modeling Checkpoint Assignment 4: Abstraction and Modeling
Week 5 Teaching CS/CT Block Programming	Walking through Checkpoint Assignments 4 and 5	Module 9 - "Do's" and "Don'ts" of Teaching CS and Computational Thinking Module 10 - Block Programming Overview	Checkpoint Assignment 5: Evaluation Lesson Plan Discussion Forum
Week 6 Block Programming	Code-along session to get started with Scratch	Module 11 - Basic Concepts for Scratch Programming Module 12 - Flow Control in Scratch	Foundational Knowledge Capstone Block Programming Discussion Forum Quiz on Basic Concepts for Scratch Programming Quiz on Flow Control in Scratch
Week 7 Block Programming	Additional Scratch code-along session Live debugging of any issues with Scratch programming project	Module 13 - Graphical and Audio Elements in Scratch Module 14 - Scripts and Subroutines in Scratch	Quiz on Graphical and Audio Elements in Scratch Quiz on Scripts and Subroutines in Scratch Scratch Programming Project
Week 8 Teaching CS/CT Block Programming	Prep for the summer CSSA courses Q&A session: Ask questions about assignments before the cut-off date The Q&A session will NOT be recorded	Module 15 - Tips for Teaching Scratch	Block Programming Lesson Plan

Mapping to California K-12 Computer Science Standards

Grade Level	Foundational Knowledge & Lesson Plan Resources sections	Block Programming section
K-2	K-2.AP.10 – Algorithms for tasks K-2.AP.11 – Model how data is stored (variables) K-2.AP.13 – Decomposition K-2.AP.14 – Develop plan K-2.AP.16 – Debug algorithm / program	K-2.AP.12 – Create program w/ loops K-2.AP.14 – Develop plan for program K-2.AP.16 – Debug algorithm / program K-2.AP.17 – Describe programming steps / choices
3-5	3-5.AP.10 – Compare / refine algorithms 3-5.AP.13 – Decomposition 3-5.AP.17 – Test / debug algorithm / program 3-5.AP.19 – Describe choices made	3-5.AP.11 – Create program w/ data (variables) 3-5.AP.12 – Create program w/ loops, events, conditions 3-5.AP.15 – Plan program w/ input from others 3-5.AP.17 – Test / debug algorithm / program 3-5.AP.18 – Peer programming / collaboration 3-5.AP.19 – Describe choices made 3-5.IC.21 – Accessibility concerns for technology
6-8	6-8.NI.5 – Explain potential security threats (optional module on cybersecurity) 6-8.AP.10 – Flowcharts and pseudocode 6-8.AP.13 – Decomposition 6-8.AP.15 – Incorporate diverse perspectives / societal need 6-8.AP.17 – Systematically test w/ test cases 6-8.AP.18 – Work collaboratively	6-8.DA.9 – Test / analyze effects of changing variables 6-8.AP.11 – Create clearly named variables w/ operators 6-8.AP.12 – Create program w/ compound flow control 6-8.AP.14 – Create procedures w/ parameters 6-8.AP.15 – Incorporate feedback from team members 6-8.AP.17 – Systematically test w/ test cases 6-8.AP.18 – Work collaboratively 6-8.IC.21 – Consider issues of bias and accessibility in design 6-8.IC.22 – Collaborate w/ many contributors and incorporate diverse perspectives
9-12	9-12.CS.1 – Describe abstraction 9-12.DA.11 – Refine computational models 9-12.AP.12 – Design algorithms 9-12.AP.13 – Create generalized solutions 9-12.AP.16 – Decomposition 9-12.AP.20 – Evaluate artifact & refine 9-12.AP.22 – Document and justify decisions 9-12.IC.25 – Application of algorithm to different disciplines	9-12.AP.14 – Justify using specific control structures 9-12.AP.15 – Design project for practical use, personal use, or to address societal issue 9-12.AP.17 – Use modular design 9-12.AP.18 – Design for broad audience w/ user feedback 9-12.AP.21 – Team programming and collaboration 9-12.AP.22 – Document and justify decisions 9-12.IC.24 – Identify impacts of bias / equity deficits on design

Academic Integrity Policy

You may discuss the assignments with others in the class, such as on the CSSA Discord channel or on the Moodle discussion forums. However, all assignments in this course are individual assignments. For individual assignments, each student must turn in their own work i their own words; no direct copying from any source or each other is allowed.

For example, you cannot copy-and-paste from a website or copy another student's submission, but you can refer to that website and summarize what you've learned, or summarize your discussion with the other student. I even encourage you to add questions you still hav particularly to the checkpoint assignments, and I'll customize my grading comments to answer those questions, as long as you submit you initial assignment before the feedback deadlines given below.

In particular, everyone should be coming up with their own solutions for the Foundational Knowledge Capstone and the Block Programming Lesson Plan. Even if you've chosen a similar topic or lesson plan, your solution should be unique to you, because you will be bringing you own unique experiences and perspectives to the problem. If I see a similar structure between multiple students' submissions, will return those for revision, with a more in-depth discussion about bringing your own creativity to the process.

Refer to the Academic Integrity policy in the campus catalog and class schedule for more details. You can also refer to the Academic Integrity policy at the Dean of Students Office at https://www.csub.edu/osrr/

Academic Accommodations

To request academic accommodations, please contact the Office of Services for Students with Disabilities (SSD) and email me an accommodations letter from the SSD Office. Policies from the SSD Office relating to accommodations, such as scheduling policies for usin their testing center, must also be followed. For more information about the services and policies of the SSD Office, contact their staff by email and/or visit their website at https://www.csub.edu/ssd/

Grading Categories

Category	Weight
Discussion participation	20%
Scratch programming project	10%
Assessment quizzes	10%
Checkpoint assignments	20%
Foundational knowledge capstone	20%
Block programming lesson plan	20%

Moodle will show your "grade to date", where unsubmitted and/or ungraded assignments will not count against your grade, until May 15, 2023.

On May 15, 2023, Moodle will be reconfigured to count unsubmitted/ungraded assignments as a 0 grade, so you know what you need to submit before the end of the term.

Description of Grading Categories

The grading categories are as follows:

- Discussion Participation: There are discussion boards on Moodle where you will need to create or participate in the indicated number o threads per discussion board for points in this category.
- Scratch Programming Project: You will program a Scratch project of your own choosing for this assignment, to show that you've learne how to program in Scratch. Extending and adapting public Scratch projects is allowed for this assignment (the "Modify" stage of the "Use-Modify-Create" model for computing).
- Assessment Quizzes: The Foundational Knowledge and Block Programming parts of the course have assessment quizzes associated wit some modules. You must score a 60% or higher to move on to the next module that has a quiz as a prerequisite. You can also attempt the quizzes as many times as you like and the highest score will be used for your grade.
- Checkpoint Assignments: These assignments are linked to Modules 3 through 7 in the Foundational Knowledge section. They are intended to slowly build you up to complete the Foundational Knowledge Capstone, by having you work on one section of the capstor report at a time. This provides you with opportunities to get feedback from me on how to improve that section of the report prior to th due date for the capstone assignment.
- Foundational Knowledge Capstone: This project has you create a simple "app" purely on paper (no programming involved) that walks through all of the stages of computational thinking to develop the "app". This is similar to the problem description, project specificatio and pseudocode stages of an actual programming project.
- Block Programming Capstone: This assignment has you adapt a lesson plan for use in your classroom (the "Modify" stage of the "Use-Modify-Create" model for computing). The lesson plan must teach one aspect of programming using either an unplugged method or Scratch, depending on what you feel is best for your students. You can use a lesson plan from your district, that you have found online, from either the Lesson Plan Resources section or Block Programming modules as the basis for your submission.

Deadlines

There are no late penalties in this course, since this course is self-paced. Each quiz and assignment has been programmed with a recommended due date to help you keep on-track, but there is no penalty if you submit the assignment or complete the quiz after the du date and before the cut-off date.

There are the following deadlines for feedback and cut-off dates:

- Friday April 28th: This is the deadline for submitting your checkpoint assignments early to get feedback prior to the deadline for the Foundational Knowledge Capstone. You must turn the checkpoint assignments in by 11:59pm on Friday April 28th to receive feedback prior to submitting the capstone assignment. Checkpoint assignments may be submitted after this date, but those submissions are not guaranteed to be graded prior to the capstone assignment deadline.
- Friday May 5th: This is the deadline for submitting your Foundational Knowledge Capstone and Block Programming Lesson Plan assignments early to get feedback prior to the cut-off date. Capstone assignments submitted by 11:59pm on Friday May 5th will be graded prior to the cut-off date. Assignments submitted by this deadline will have the option to revise and resubmit prior to the cut-of date for regrading if you wish to improve your grade.
- Monday May 22nd: This is the cut-off date for all submissions. All assignments must be turned in by 11:59pm on May 22, 2023 so I hav sufficient time to grade assignments for all of my Spring courses before grades are due on May 24, 2023.

Please keep my workload in mind as well and try to submit your assignments in advance of these dates.

Changelog

19 Mar 2023 - Updated wording on California K-12 standards

Last modified: Sunday, March 19, 2023, 12:14 PM

◄ CA Board of Education: K-12 CS Outcomes and Standards

Jump to...

Zoom link for SCI 5200 on Mondays at 4:00pm and Office Hours on Wednesdays at 4:00pm ►