CPSC 120
Principles of Computer Science

What is Computer Science?

- the study of the principles and use of computers
- on the theoretical side
  - models of computation
  - what problems can be solved with a given model of computation
  - what problems can be solved efficiently with a given model of computation
- on the applied side
  - designing algorithms to solve problems in a variety of domains
  - producing software
- not about how to use programs to accomplish tasks
- not just programming – about what programs can do

This Course

- will introduce computer programming and the core principles of computer science in the context of creating interactive animated “sketches”
  - art
  - illusions
  - animations
  - games
  - fractals
  - simulating the natural world
  - image manipulation
- will use the programming language Processing
  - designed specifically for non-programmers who want to create programs using graphics, images, animation, and interaction
  - it's easy to do cool stuff! (but it's also Java underneath)

Why Study Computer Science?

- because you are interested in learning to write programs
- because you are interested in solving problems with computation
  - aspects of computing – algorithms, programming, hardware
  - domains – operating systems, compilers, networks, databases, artificial intelligence, robotics, graphics, …
- because you want to learn skills useful in a variety of areas
  - decomposition and abstraction
  - algorithmic thinking, structured logic, and problem solving
  - software literacy – computers and applications of computing are everywhere

Why Take This Course?

- because you are interested in computer science
- because you *might* be interested in computer science
- because you want to expand your creative practice in the visual arts to include programming
- because being able to create your own programs expands your toolbox in other fields

CPSC 120 vs 124

CPSC 120 is intended for

- non-majors
- students considering a major or minor in CS
- students without previous CS or programming background

You should take CPSC 124 instead if you

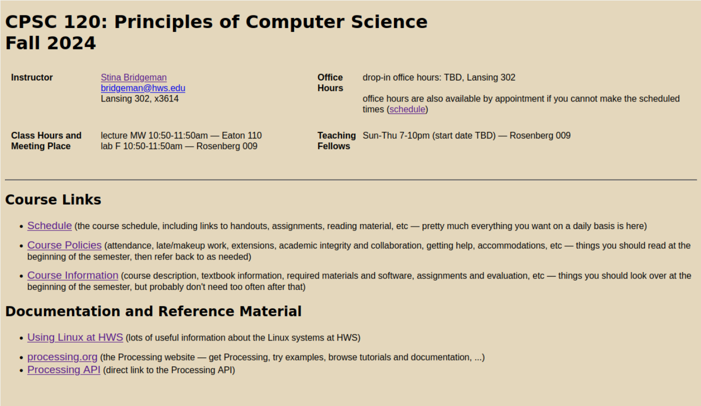
- have CS or programming experience
- know you are likely to major or minor in CS
- specifically want to learn to program

CPSC 120 will cover many of the basic programming concepts as CPSC 124, but not in the same depth.

- you won't become an expert programmer with just this course
- you will get a head start on CPSC 124, and will have a foundation for learning more about programming

Course Website

http://math.hws.edu/bridgeman/courses/120/f24/



CPSC 120: Principles of Computer Science
Fall 2024

Instructor	Stina Bridgeman bridgeman@hws.edu Lansing 302, x3614	**Office Hours**	drop-in office hours: TBD, Lansing 302 office hours are also available by appointment if you cannot make the scheduled times (schedule)
Class Hours and Meeting Place	lecture MW 10:50-11:50am — Eaton 110 lab F 10:50-11:50am — Rosenberg 009	**Teaching Fellows**	Sun-Thu 7-10pm (start date TBD) — Rosenberg 009

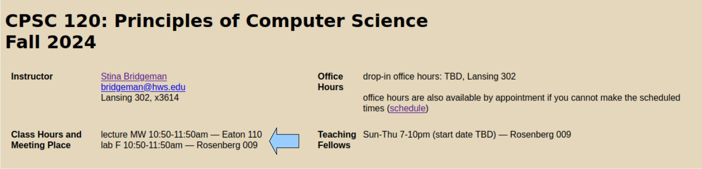
Course Links

- Schedule (the course schedule, including links to handouts, assignments, reading material, etc — pretty much everything you want on a daily basis is here)
- Course Policies (attendance, late/makeup work, extensions, academic integrity and collaboration, getting help, accommodations, etc — things you should read at the beginning of the semester, then refer back to as needed)
- Course Information (course description, textbook information, required materials and software, assignments and evaluation, etc — things you should look over at the beginning of the semester, but probably don't need too often after that)

Documentation and Reference Material

- Using Linux at HWS (lots of useful information about the Linux systems at HWS)
- processing.org (the Processing website — get Processing, try examples, browse tutorials and documentation, ...)
- Processing API (direct link to the Processing API)

Class Meetings

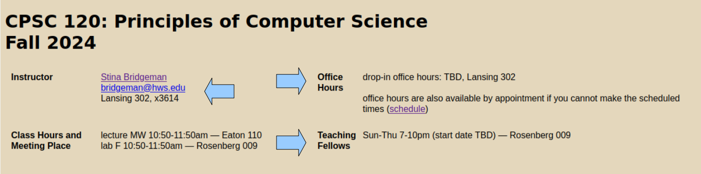


CPSC 120: Principles of Computer Science
Fall 2024

Instructor	Stina Bridgeman bridgeman@hws.edu Lansing 302, x3614	**Office Hours**	drop-in office hours: TBD, Lansing 302 office hours are also available by appointment if you cannot make the scheduled times (schedule)
Class Hours and Meeting Place	lecture MW 10:50-11:50am — Eaton 110 lab F 10:50-11:50am — Rosenberg 009	**Teaching Fellows**	Sun-Thu 7-10pm (start date TBD) — Rosenberg 009

- class meets MWF 10:50-11:50
- (most of) Friday’s classes are lab sessions, held in Rosenberg 009
- two Wednesdays are also in Rosenberg 009 – see the schedule page

Office Hours and Teaching Fellows



CPSC 120: Principles of Computer Science
Fall 2024

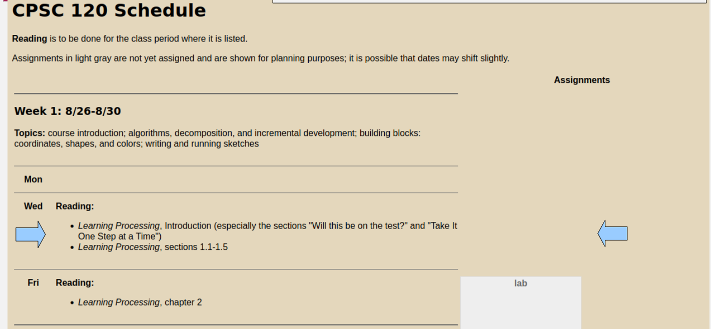
Instructor	Stina Bridgeman bridgeman@hws.edu Lansing 302, x3614	**Office Hours**	drop-in office hours: TBD, Lansing 302 office hours are also available by appointment if you cannot make the scheduled times (schedule)
Class Hours and Meeting Place	lecture MW 10:50-11:50am — Eaton 110 lab F 10:50-11:50am — Rosenberg 009	**Teaching Fellows**	Sun-Thu 7-10pm (start date TBD) — Rosenberg 009

- office hours are drop-in – no appointment necessary
  - held in Lansing 302
- email questions and/or make an appointment if you can’t attend office hours

- Teaching Fellows are CS majors
- they haven’t taken this course, but they can provide guidance with how to approach trying to solve problems
- TF hours are held in Rosenberg 009

Schedule Page

check here for readings, assignments, handouts, examples from class, etc



CPSC 120 Schedule

Reading is to be done for the class period where it is listed.

Assignments in light gray are not yet assigned and are shown for planning purposes; it is possible that dates may shift slightly.

Assignments

Week 1: 8/26-8/30

Topics: course introduction; algorithms, decomposition, and incremental development; building blocks: coordinates, shapes, and colors; writing and running sketches

Mon

Wed **Reading:**

- *Learning Processing*, Introduction (especially the sections "Will this be on the test?" and "Take It One Step at a Time")
- *Learning Processing*, sections 1.1-1.5

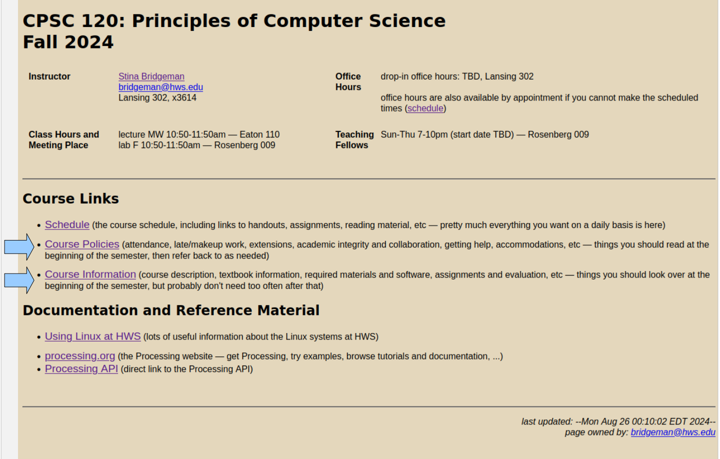
Fri **Reading:**

- *Learning Processing*, chapter 2

lab

homework –
- reading for Wednesday
- intro survey (to be completed this week)

Important Policies and Other Information



CPSC 120: Principles of Computer Science
Fall 2024

Instructor	Stina Bridgeman bridgeman@hws.edu Lansing 302, x3614	**Office Hours**	drop-in office hours: TBD, Lansing 302 office hours are also available by appointment if you cannot make the scheduled times (schedule)
Class Hours and Meeting Place	lecture MW 10:50-11:50am — Eaton 110 lab F 10:50-11:50am — Rosenberg 009	**Teaching Fellows**	Sun-Thu 7-10pm (start date TBD) — Rosenberg 009

Course Links

- Schedule (the course schedule, including links to handouts, assignments, reading material, etc — pretty much everything you want on a daily basis is here)
- Course Policies (attendance, late/makeup work, extensions, academic integrity and collaboration, getting help, accommodations, etc — things you should read at the beginning of the semester, then refer back to as needed)
- Course Information (course description, textbook information, required materials and software, assignments and evaluation, etc — things you should look over at the beginning of the semester, but probably don't need too often after that)

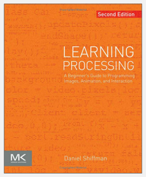
Documentation and Reference Material

- Using Linux at HWS (lots of useful information about the Linux systems at HWS)
- processing.org (the Processing website — get Processing, try examples, browse tutorials and documentation, ...)
- Processing API (direct link to the Processing API)

last updated: --Mon Aug 26 00:10:02 EDT 2024--
page owned by: bridgeman@hws.edu

Course Materials

- textbook is *Learning Processing* by Daniel Shiffman, 2nd ed
  - chapter 1 is available at learningprocessing.com


Second Edition
LEARNING PROCESSING
A Beginner's Guide to Programming Images, Animation, and Interaction
MK
Daniel Shiffman

- all of the necessary software is available –
  - on the lab machines in Rosenberg 009 and Lansing 310
  - remotely through the Linux VDI
  - it can also be downloaded for free so you can set up your own computer (optional)
- ☆ if you have a laptop that you can bring to class, please do so (starting Monday)
  - also bring to lab on Friday to get things set up

Expectations

- attend all scheduled class and lab sessions
- ☆ attend at least 5 hours of office hours and/or Teaching Fellows over the course of the semester
  - 5 hours = e.g. 1 hour of TFs every 3 weeks or a 20-minute office hours visit every week
  - can also count relevant department and campus talks – requires attendance plus a short writeup
- spend approx. 8 hours per week outside of class on reading, assignments, and studying
  - if you routinely spend less time, you may not be mastering the material – challenge yourself with extra credit if you are completing the assignments
  - if you routinely spend more time and feel like you are stuck and not making progress, come to office hours

Assignments and Evaluation

- reading – first exposure to the material
  - try the exercises in the book to help test your understanding
    - it is OK if you don’t understand everything, but you should attempt to understand things and be prepared with questions about what you don’t
  - ☆ reading is due the day it is listed
- labs [34%] – opportunity for practice
  - started in Friday lab sessions, due one week later
- projects [20%] – practice and mastery
  - larger programs integrating multiple concepts
  - dates are on the schedule page
- exams [36%] – demonstration of mastery
  - two midterms and a final
  - written, closed book with a page of notes allowed
  - ☆ dates are on the schedule page

Assignments and Evaluation

- engagement and participation [10%]
  - learning isn’t a passive activity; engagement is required
  - be prepared for class, make meaningful contributions to class, actively contribute to group activities, be on-task during class and lab, attend office hours and/or Teaching Fellows
  - disruptive behavior, lack of participation, poor progress on in-class activities, and excessive absences will lower your participation grade
- extra credit
  - opportunities on projects and some labs
  - can also attend relevant talks (with a short writeup)

Etiquette

- arriving late, leaving early, and coming and going during class is distracting
  - please do your best to arrive on time, and to take care of any necessary business before or after class so you can stay for the whole period
  - let me know if you know in advance that you need to miss part or all of a class
- ensure that phones do not ring or generate audible notifications during class
- please do not wear headphones in lab

Policies

- there is a steady workload
- foundational topics introduced earlier are used and built on later
- late handins take away time for the next assignment and can quickly snowball into falling behind – a late handin or extension should be rare

- attendance – attending class/lab should be a high priority
  - if you must miss class, check the schedule page for missed material and new assignments – it is your responsibility to catch up promptly
  - missing more than six classes/labs will lower your participation grade
  - missing 4-6 classes/labs will lower your participation grade unless you are proactive about communication, diligent about catching up, and more engaged in other aspects of the course
- late policy
  - 10% per day up to a maximum of 30%
  - maximum of one week late
- extensions
  - ⚠ an "automatic" two-day extension can be requested *before the due date* for any assignment, no reason required
  - if there are circumstances outside your control which are significantly impacting your ability to focus on academics for several days or more and a two-day extension is insufficient, reach out ASAP
- rescheduling exams – being in class on exam day should be a high priority
  - if you have an unavoidable conflict with the day of an exam, *notify me as soon as possible* and make arrangements *in advance*
  - a missed exam can only be made up after the fact in the class of a last-minute emergency that prevents you from attending class – notify me and make makeup arrangements as soon as possible

Policies

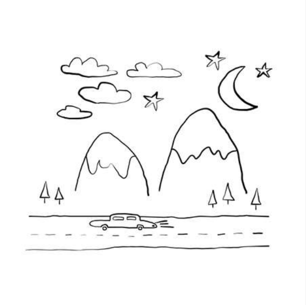
- academic integrity and collaboration
  - ☆ this course is about the *process* of producing programs as well as the technical content of a working program
    - while the program is the obvious thing that is handed in, the important lesson is the development of that program
  - ☆ what *you* turn in for a grade must reflect *your* engagement with this process
    - always make the first attempt at a solution yourself, using only the materials from class (textbook, slides, examples)
    - utilize office hours and Teaching Fellows for help
    - see the full collaboration policy on the webpage
  - a broader range of sources is OK for ungraded practice, but –
    - be aware of the reliability (or lack thereof) of your sources
    - be careful to avoid over-reliance

Being Successful

- programming is hard!
  - it is likely a new way of thinking
  - you must get every detail exactly correct, all at the same time
  - you will most likely get stuck! (this is not a sign of failure or lack of ability)
- start assignments early so you have opportunities to ask questions and get help
- utilize office hours and the Teaching Fellows
  - don't by shy!
  - remember that there is the expectation of attending at least 5 hours over the course of the semester
- stay on top of the material – things can pile up quickly
  - reach out if you are stuck, struggling with the material, or feeling overwhelmed

Exercise

- work in pairs
- one person faces the screen, the other has their back to the screen
- round 1 – the person facing the screen gives instructions for drawing the scene to the other, who draws it
  - the drawer does not show what they are drawing to the describer!
  - the goal is for the drawer to accurately reproduce the scene without being able to see the picture



Discussion

- How successful was the drawer in recreating the picture?
- What kinds of things did the describer say to draw?
- What did the drawer already need to know in order to follow the instructions?
- Were the instructions detailed enough? Did the drawer have to make his or her own decisions?
- How did the describer tackle describing a scene with many elements?

Algorithms and Programs

- an *algorithm* is a list of instructions for carrying out a task
  - sequential – must capture the order in which the instructions are to be performed
  - detailed – computer has no knowledge with which to fill in gaps
  - precise – computer cannot figure out what you mean if instructions are ambiguous
- a *program* is an algorithm expressed in a particular programming language
  - can be executed by a computer
  - provides certain kinds of instructions

Building Blocks, Patterns, and Decomposition

- need a shared vocabulary for drawing
  - a set of *primitives* – basic shapes (e.g. rectangle, ellipse) used to construct more complex visuals
  - a notion of *space* to express position and size
- *patterns* provide templates for recognizing similar tasks and applying known solutions
  - makes algorithm and program development easier!
- *decomposition* refers to breaking bigger tasks down into smaller pieces
  - important for making problem solving and algorithm development easier – small problems are simpler than big ones!

Exercise

- round 2 – switch roles and repeat
  - describer, keep in mind the ideas of a notion of space, primitives, and decomposition
  - drawer, show what you are drawing to the describer after each instruction

Discussion

- Does the describer getting feedback make things easier?

Incremental Development

- *incremental development* makes creating a correctly working program easier
  - implement the program in stages, testing after each bit
    - if something doesn't work, focus on the part(s) added since the last test