Structure and Interpretation of Computer Programs

This is intended to be a definitive resource on self-teaching the classic Computer Science book, **Structure and Interpretation of Computer Programs**, commonly SICP.
This page compiles lectures and readings from Berkeley's series from 2011 (originally scattered throughout old course pages and dead links) to produce a clean path through the textbook that focuses on the content.

Prerequisites

THIS IS A HARD BOOK. Although the course allows for no former programming experience, approaching the course this way will make it needlessly difficult. Berkeley, in their version of the course recommends:

- **Maths up to Single Variable Calculus**: limits, functions, integration, differentiation.
    - Optionally, **Discrete Maths and proofs** will be extremely helpful for the course's harder exercises.
- **Some programming experience:** some confidence in using recursion is cited as a benchmark. If you're in between comfort levels, we strongly recommend the twin courses How to Code: Simple Data and Complex Data as a precursor.

How to Code: Simple Data

There is one session available: This programming course takes a unique approach, as it focuses on learning a systematic programming method rather than a programming language. This practical approach will help you channel your creativity so that you can program well in any language.

https://www.edx.org/course/how-to-code-simple-data

Course Introduction

⚠️ This page relates to the **Scheme** version of SICP, as opposed to newer versions that teach with JavaScript or Python.

What to expect (read before starting)

The course is split into 16 weeks. Materials given are in order (ie reading should precede homework should precede … etc.) Each week will have readings, homework, a lab section, and 2-3 lectures. Some intervals will have projects and midterm exams associated with them.

Readings

These are usually taken from the SICP textbook. This resource will link to the corrected 2nd edition, available free online. The textbook contains exercises, see Homework below for details. Occasionally a non-SICP handout is given - these will be linked.

Homework

From the lecturers:

> Homework assignments consist mostly of exercises from the textbook; you'll do
> these whenever you can schedule time, either in the lab or at home. You may be accustomed to textbooks with huge numbers of boring, repetitive exercises. You won't find that in our text! Each assigned exercise teaches an important point.

Because of the textbook's popularity, answers for every SICP exercise can be easily found online through Google.

Lab sections

These are quick projects, usually intended to be completed in groups. A good opportunity for collaboration if you are going through these texts alongside somebody else!

Lectures

Each week has 2 or 3 lectures associated with it. Given are lecture notes, the appropriate ones will be linked on the week's page. Keep in mind this quote from the course creators:

> The course reader includes my lecture notes. What this means is that you should be
> able to devote your effort during lecture to thinking, rather than to frantic scribbling.

Programs from the lectures can be accessed in the cs61a/lib folder of the course materials (see below)

💡 Don’t worry about the missing numbers in the lectures (there’s no lecture 15, or 27-29) - the lectures are not lost, but rather they correspond to half-terms and spring breaks.

Projects

These are larger projects for you to apply more of your skills, given every few weeks or so. Often these projects are intended for pairs.

Exams

There are three midterm exams and a final. For each, three sample papers (with answers) are given for each.

For completionists…

Not every chapter in SICP is covered in these lecture series (the last of the 5 segments is entirely dropped, for one). A full list of chapters skipped is given here:

List of chapters not covered

Every exercise is also, not necessarily assigned. But solutions for every one can be easily found online if you want to go ahead with them.

Get started

Download the materials

- To get started, clone the course materials from the github repository linked here.
- We use the online SICP given here, as created by sarabander

Structure and Interpretation of Computer Programs

Second Edition Unofficial Texinfo Format 2.andresraba6.6 Harold Abelson and Gerald Jay Sussman with Julie Sussman, foreword by Alan J. Perlis ©1996 by The Massachusetts Institute of Technology This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (). Based on a

https://sarabander.github.io/sicp/html/index.xhtml

Course calendar

Week	Reading	Homework	Lab	
1	1.1	✓	✓	Lecture 01: functional programming 1
				Lecture 02: functional programming 2
2	1.3	✓	✓	Lecture 03: higher-order procedures 1
				Lecture 04: higher-order procedures 2
3	1.2.1-4	✓	✓	Lecture 05: user interface -Alan Kay-
				Lecture 06: user interface -Alan Kay-
				Lecture 07: orders of growth
				Lecture 08: recursion and iteration
Project 1				
4	2.1, 2.2.1	✓	✓	Lecture 09: data abstraction

Week	Reading	Homework	Lab	
				Lecture 10: sequences
				Lecture 11: Example: calculator
Midterm 1				
5	2.2.2-3, 2.3.1-3	✓	✓	Lecture 12: hierarchical data
				Lecture 13: hierarchical data
				Lecture 14: Example: Scheme-1 interpr
Project 2				
BREAK				Lecture 15 is not given due to a half-term
6	2.4-2.5.2	✓	✓	Lecture 16: generic operators
				Lecture 17: generic operators
7	oop_aboveline.pdf (given)	✓	✓	Lecture 18: object-oriented programmi
				Lecture 19: object oriented programm
				Lecture 20: object-oriented programmi
Midterm 2				
8	3.1, 3.2, oop_belowline.pdf (given)	✓	✓	Lecture 21: assignment and state

Week	Reading	Homework	Lab	
				Lecture 22: environments
				Lecture 23: environments
Project 3 (a)				
9	3.3.1-3	✓	✓	Lecture 24: mutable data
				Lecture 25: mutable data
				Lecture 26: vectors
Project 3(b)				
BREAK				Lecture 27-29 is not given due to spring break
10	3.4	✓	✓	Lecture 30: client-server programming
				Lecture 31: concurrency
				Lecture 32: concurrency
11	3.5.1-3, 3.5.5, therac25.pdf (given)	✓	✓	Lecture 33: streams
				Lecture 34: streams
				Lecture 35: Therac-25
Midterm 3				
12	4.1, mapreduce.pdf (given)	✓	✓	Lecture 36: metacircular evaluator -

Week	Reading	Homework	Lab	
				Lecture 37: metacircular evaluator -
				Lecture 38: mapreduce
				Lecture 39: mapreduce
13	See homework	✓	✓	Lecture 40: analyzing evaluator
14	4.2,4.3	✓	✓	Lecture 41: lazy evaluator
Project 4a				
15	4.4.1-3	✓	✓	Lecture 42: logic programming
				Lecture 43: logic programming
Project 4b				
16	n/a	n/a	n/a	Lecture 44: Review
Final exam				

Credits

Structure and Interpretation of Computer Programs

> Harold Abelson and Gerald Jay Sussman with Julie Sussman, foreword by Alan J. Perlis
> (c) 1996 by The Massachusetts Institute of Technology

Lecture materials made generously available online by Brain Harvey and the University of California, Berkeley.

This resource compiles those already freely available and claims no ownership.