SFU Computer Science
Teaching Toolkit
Introduction

Studies have shown that students, especially women, are attracted to computer science for problem-solving as a means to an end [1]. In 2018, SFU Computing Science faculty member Dr. Angelica Lim completed a re-work of SFU’s Introduction to Computing Science and Programming (CMPT 120), incorporating problem-solving as a fundamental paradigm.

While the large majority of students belonged to non-CS majors, survey results showed that students agreed that the course helped them decide to further pursue studies in Computing Science.

“Although I have been interested in programming since high school, I was always too intimidated to try it. I am really glad to have finally taken this course because I found that I enjoyed programming and problem solving more than I expected to. I had so much fun throughout the entire semester, and will be taking more computing science courses this summer - hopefully I can pursue a minor!”

An exceptionally high 98% of students agreed or strongly agreed that the course related concepts to current issues or real-life situations. And 94% agreed or strongly agreed that the course projects and/or assignments provided opportunity for creativity and creative thinking.

The goal of the SFU Computer Science Teaching Toolkit is to provide the same exercises and tools deployed in first year university to all British Columbia teachers. These exercises and tools highlight the utility and breadth of computer science while teaching fundamental computer science concepts and programming constructs.

SFU Computer Science Teaching Toolkit (SFU CSTT)

The learning outcomes for this toolkit are equivalent to How To Think Like a Computer Scientist (Ch. 1-12,15), an open educational resource increasingly used for introductory CS courses in high school and post-secondary. Students completing the curriculum should be able to pass the SFU placement test to enter into CMPT 125.

Importantly, the SFU Computer Science Teaching Toolkit teaches all concepts through engaging examples. Ordering is aligned with exploration units relevant to today’s technology-driven world: Chatbots, Computer Vision, Internet and Big Data, Recommendation Systems, and more. Strings are used early on as the main building block for lists, functions, etc., revealing the use of Computer Science in many diverse areas and fields. Additionally, code comprehension exercises, traditionally containing terse, single-letter variables, are re-written in equivalent code with inclusive and approachable variable names.

Exploration Units and Exercises

The teaching toolkit contains 29 interactive lessons containing the exercises listed below. Each lesson was completed during a 50-minute university block, which can be extended to longer lesson times for high school students.

Unit 0 - Introduction to Computer Science
In this first unit, you'll learn what Computer Science is, where it came from, and start to learn about how it works. You'll get set up with the tools that we'll be using in this class to help you become proficient computer scientists by the end of this course.

- Motivational Quote Generator

CS Topics: Algorithms, comments, output

Unit 1 - Chatbots
You will be learning about the basics of programming through exercises loosely based on chatbots, like Amazon Alexa or Siri. You'll make your first program in Python, learn about outputting to the screen, creating lists, working with text, and changing output depending on conditions.

- Motivational Quote Generator
- Greetings Chat Bot
- Horoscope Bot
- Fortune Cookie Generator
- Food Bot
- Mindreader Game
- New Year’s Bot
- Star Wars Bot

CS Topics: Variables, concatenation, input, lists, random module, conditionals, boolean statements, error types, robustness, string methods, in keyword, chaining functions, for loops, lists from input

Unit 2 - Recommendation Systems
Recommendation systems such as Netflix's and Amazon's "You might also like" features are a great way to help people discover new things they may like. Under the hood, it's statistics, so we'll show you how to do that in a Python program.

- Popular Cafe Finder
- Chip Rater App
- City Bliss Evaluator
- Weight Calculator
- Age in 2049
- Olympic Judging
- Favourite Spots
- Comparing Favourite Movies
- Similar People Finder
- McDoland’s
- Parent Bot
- Similar Hobby Finder

**CS Topics:** Avoiding code duplication, loops with range, accumulator variables, string/int/float data types, type conversion, length, division, order of operators, split to lists, indexing in lists, reading/opening files, comparison operators, comparing lists, similarity measures, nested loops

**Unit 3 - Under the Hood**
We've learned about many applications of computer science, but what's happening in our machines to make this happen? During this short module, we'll explore deeper into the machine's binary code to understand how it all works.

- Binary to Decimal Converter
- Binary Bit Bot

**CS Topics:** Binary, ASCII, Unicode, Bytes

**Unit 4 - Graphics and Animation**
Pixar movies and your favourite 3D animated films these days are built with code. You'll make a basic animation using code, and learn how to build complex and beautiful graphics using a technique called recursion.

- Interactive Drawing Turtle
- Drawing a Cookie
- Palindrome Checker
- Recursive Tree Challenge
- English<>French Translator
- Power Function
- Input Validator
- Places to Visit
- Forest Surveyor Bot
- Number Guessing Game

**CS Topics:** Turtle module, while loop, functions, parameters, recursion, dictionaries, fruitful functions, while (advanced)

**Unit 5 - Computer Vision**
Computers are so smart that they can recognise faces in images, and you can even unlock your phone with your face! How do they do that? And how can cameras be used to guide robots and
autonomous cars? We'll talk about how that works, and you'll write Python code to process and understand photos.

- Combining Images with Green Screen
- Cool Colours Module
- Light-coloured Pixels
- Image to Black & White
- Jellybean Counter (with timer)
- Robot Puppy

CS Topics: Importing with from ... import, 2D arrays, tuples, RGB pixel representations, nested loops (cont'd), making modules, append, problem solving strategies, string formatting for output, efficiency, time module

Unit 6 - Internet and Big Data
The Internet has become a hub of information. Lots of information. How can we sift through these billions of web pages and information points efficiently? We'll talk about searching and sorting algorithms, how to evaluate their effectiveness at the scale of millions.

- Top Danceable Songs (Spotify music dataset)
- Sorting Cards
- Sort By Score
- Sum of Lists and Factorial
- Binary Search on List of Chapters

CS Topics: List indexing (cont'd), linear search, selection sort, complexity, selection sort (cont'd), swapping, range sublisting, list slicing, binary search, Big O notation, recursion (cont'd)

Listing of Assignments
Students were given 1.5 days to complete their homework assignments and extend their program to the best of their ability.

- Chatbot with Personality
- Chatbot with a Loop
- Calculate this, Computer!
- Comparing Things
- Make an Interactive Drawing or Animation
- Practice, practice, practice (Recursion, Functions, Dictionaries)
- Image Magic
- Image Understanding
- A Smart Music Bot
Listing of Exam Questions

Test #1: Exploration Units 0, 1 and 2  
Covering 13 lessons  
Time allotted: 50 min x 2

- Multiple choice Theory and Comprehension test (14 questions)  
- Coding test (2 questions)

Test #2: Exploration Units 3 and 4  
Covering 6 lessons  
Time allotted: 50 min x 2

- Multiple choice Theory and Comprehension test (14 questions)  
- Coding test (2 questions)

Final exam: Comprehensive, with emphasis on Exploration Units 5 and 6  
Covering 10 lessons  
Time allotted: 1 hour x 2

- Multiple choice Theory and Comprehension test (15 questions)  
- Coding test (3 questions)

Coding questions include:

- World Traveller Bot  
- Semester Evaluator Bot  
- Souvenir Shopping Bot  
- Mirror Rorrim  
- Unique Baby Animal Names  
- Old MacDonald Had a Vowel  
- Wine Snob

Code Comprehension Examples

- What does this code output?  
- What does this code output if the input is __?  
- Is the following code valid?  
- What is the complexity of the following code?
Guiding principles

1. **Make parallels with foreign language learning.** We acknowledge that CS can feel difficult at first because learning any new language, like Chinese or French, takes time and practice. But once you become fluent, you can translate from English in your head, and communicate with a new friend - the computer!

2. **Write useful code from Day 1.** Our approach takes concepts and explores a useful, end-to-end program each lesson.

3. **Introduce 1 or 2 concepts at a time, in context.** Sample programs are carefully crafted so that students are not overloaded with multiple concepts at a time. And just like learning natural language, new concepts are introduced embedded in real programs and situations, not in isolation.

4. **Design algorithms in English.** Thoughts and form are two different processes, especially at first. We make it a best practice to design the algorithm flow in English comments first, then translate into Python.

5. **Handle strings first.** We reduce cognitive overload by exploring string manipulation and all its possibilities, first.

6. **Construct with creativity.** Exercises are open-ended rather than looking for a right answer. We reinforce from the beginning the idea that programming is a creative endeavour.

7. **Share code with others.** We make it a delight to build programs and share them with friends and family. We use an online coding space like repl.it or trinket.io so that students can take pride in their work and have others test them out!

8. **Focus on fluency.** Our ultimate aim is to develop students who can code. We deliver test questions that ask students to write full programs from scratch.

9. **Write beautiful, understandable code.** Obscure, tricky code is often used for tests of understanding, but we ensure that students are aware that real code written by professionals is highly readable, and uses descriptive variable names. No longer are variables limited to i, j and k!

10. **Showcase fields in CS.** Students learn coding through exploration units in contemporary fields in CS, such as computer vision, natural language processing, and recommendation systems.

---

**Author and Contact Information**

**Dr. Angelica Lim** is a roboticist, educator and champion for diversity in computer science. She was the founding president for the SFU Women in Computing Science society in 2003, and Director of the SFU Invent the Future summer program for girls to discover the field of artificial intelligence. Originally from Coquitlam, she received her B.Sc. in Computing Science and French at SFU, and Master's and Ph.D. in Computer Science at Kyoto University, Japan. Previously, she worked as a Software Engineering Manager at SoftBank Robotics in Paris. She is currently an Assistant Professor of Professional Practice in Computing Science at Simon Fraser University.

---

To receive free access to exercises and test questions, visit: [www.sfu.ca/computing/cstt](http://www.sfu.ca/computing/cstt)