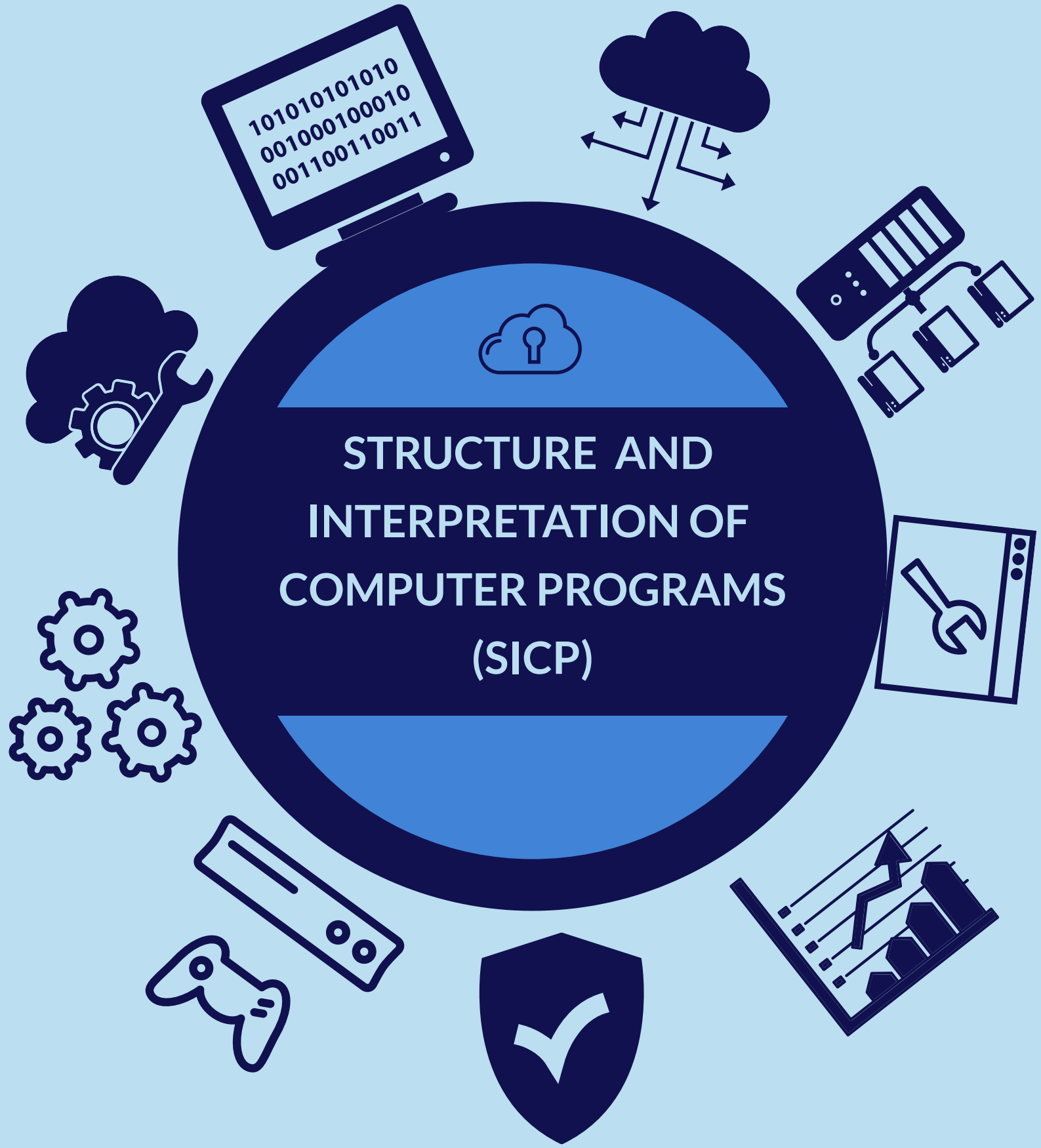
NUS | Computing
National University of Singapore



STRUCTURE AND INTERPRETATION OF COMPUTER PROGRAMS (SICP)

WHAT IS SICP?

The Structure and Interpretation of Computer Programs (SICP) is an online course which provides a thorough introduction to computational thinking. Starting from a small core of fundamental abstractions, the programme introduces programming as a method for communicating computational processes. The programme begins with purely functional programming, using a simple substitution-based execution model, and ends with a powerful modern imperative language and a realistic environment-based execution model.

Topics covered include: functional abstraction, recursion, higher-order functions, data abstraction, algorithmic strategies, state mutation, loops and arrays, evaluation strategies and symbolic processing.

Non-CS students benefit from this introduction to computational thinking by gaining a deep appreciation for computation, a mode of inquiry of ever increasing importance. Students will be able to assess what kinds of problems are amenable to computational approaches and, in principle, how computational solutions are constructed. The material is building on the students' understanding of formal processes in middle and high school science and mathematics.

In the practical exercises, students will be using a sublanguage of JavaScript called Source, see https://en.wikipedia.org/wiki/Source_(programming_language).

WHO IS IT FOR?

Undergraduates with no or only casual prior exposure to computation and programming.

RESOURCES

TEXTBOOK

The course uses the textbook "*Structure and Interpretation of Computer Programs, JavaScript Adaptation*" (SICP JS), which is the modernized edition of the classic textbook "*Structure and Interpretation of Computer Programs*".

SICP JS will be published by MIT Press in 2022. This will be the first time that the complete and print-ready version of this textbook will be used in a course, world-wide.

Here is the link to the web edition of the textbook: https://source-academy.github.io/sicp

SOURCE ACEDEMY

The Source Academy is an award-wining computer-mediated learning environment for studying the structure and interpretation of computer programs. Students write and run their programs in their web browser, using sublanguages of JavaScript called Source, designed for the textbook Structure and Interpretation of Computer Programs, JavaScript Adaptation.

The Source Academy is available under the Apache License 2.0 at our GitHub organisation, Source Academy. The National University of Singapore uses the Source Academy for teaching Programming Methodology to freshmen Computer Science students in the course CS1101S.

Check out Source academy here: https://sourceacademy.nus.edu.sg/

BENEFITS

NUS ACCOUNT

exclusively created for you with access to **NUS LEARNING PLATFORMS** during the programme

SYNCHRONOUS LEARNING

via **ZOOM** with real-time social interaction for discussion, feedback, sharing and insights

STUDIO SESSIONS

with at most **8 STUDENTS** per instructor to provide optimal individualised learning experience

PRACTICAL EXAMPLES

provided to develop concrete understanding guided by instructor well-versed in the area

CERTIFICATE

AWARDED FOR EACH UNIT COMPLETED which can be greatly advantageous for future career and studies

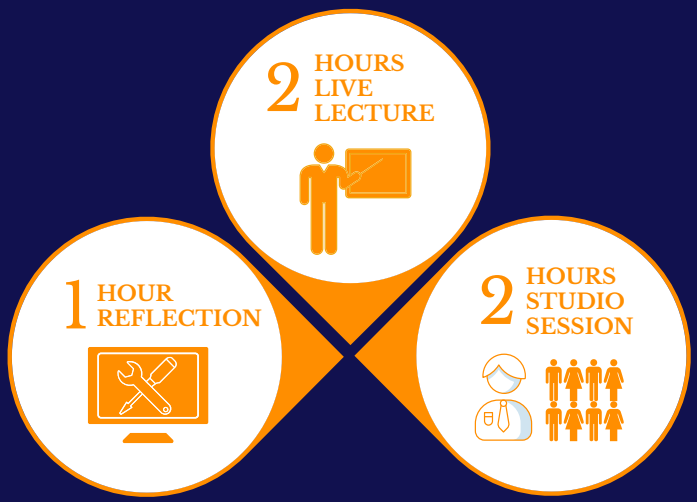
ADVANTAGE IN MCOMP (GENERAL TRACK) APPLICATION

offered by NUS School of Computing for students who perform well in SICP

For more information on MComp (General Track):
https://www.comp.nus.edu.sg/programmes/pg/mcomp-gen/

CONTACT HOURS

Each instructional day consists of 5 contact hours, including 2 hours live lecture, 1 hour reflection and 2 hours studio session.



STRUCTURE

SICP consists of 4 units conducted over 3 different periods throughout the year. Students will first register and pay for Unit 1. Upon passing Unit 1, students will be informed to proceed to and pay for Unit 2 & 3, which will be taught in a consecutive manner. Finally, upon passing Unit 2 & 3, students will be informed to proceed to and pay for Unit 4.

Students will be awarded a certificate upon completion of each Unit. Upon completion of all 4 Units, students will receive an overall programme completion certification.

UNIT 1

BUILDING ABSTRACTIONS WITH FUNCTIONS

8 Instructional Days on Saturdays & Sundays

between

8 May 21 - 5 Jun 21

UNIT 2

BUILDING ABSTRACTIONS WITH DATA

UNIT 3

IMPERATIVE PROGRAMMING

12 Instructional Days

between

12 Jul 21 - 28 Jul 21

UNIT 4

BEYOND CONVENTIONAL PROGRAMMING

6 Instructional Days on Saturdays & Sundays

between

4 Dec 21 - 2 Jan 22

UNIT INTRODUCTION

UNIT 1: BUILDING ABSTRACTIONS WITH FUNCTIONS

In this unit, students are introduced to a high-level mental model for computation that builds on formula manipulation in mathematics and science.

Students will learn that programming with functions can be understood as describing computational processes in which each step consists of an expansion or simplification that follows simple rules. With this understanding, students can build solutions to computational challenges in computer graphics.

Learning Objectives

- Functional Programming
- Substitution Model
- Scoping and Names
- Recursion
- Higher-Order Function

UNIT 2: BUILDING ABSTRACTIONS WITH DATA

Learning Objectives

- Data Abstraction
- List and Tree Processing
- Symbolic Processing
- Sorting and Searching in Lists

In this unit, students will understand data as an abstraction technique for handling information. They will form a mental model for data structures that connects to the computational model of Unit 1.

Students will learn how to systematically manipulate data structures with the programming abstractions provided by functional programming. The example domains range from sound processing to symbolic data processing.

UNIT INTRODUCTION

UNIT 3: IMPERATIVE PROGRAMMING

In this unit, students will learn the expressive power of modern, high-level programming languages and discover the environment model as a mental model for understanding the computational processes arising in those languages.

The students will learn programming with conventional control structures such as loops and with conventional data structures such as arrays. The example domains in this unit include robotics and video processing.

Learning Objectives

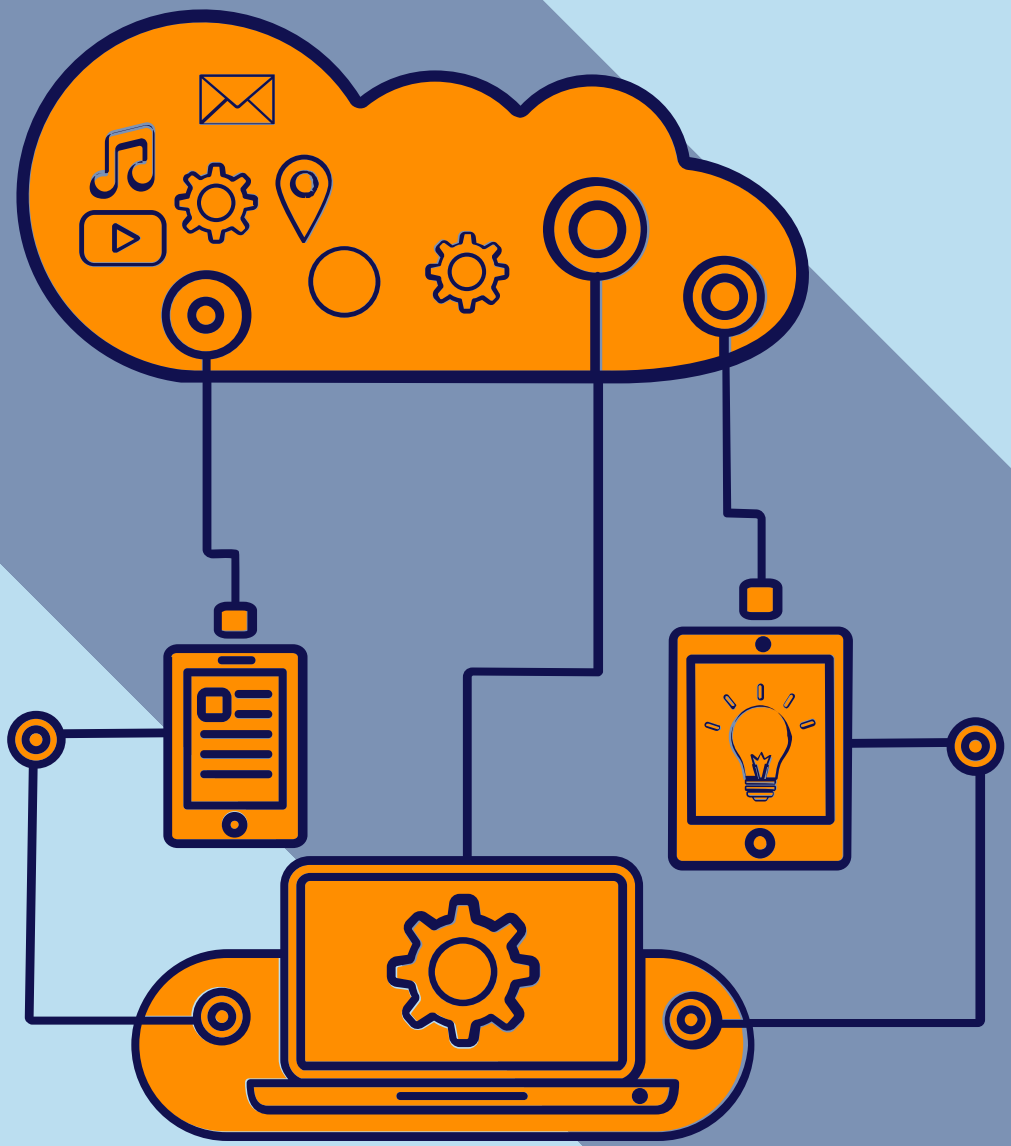
- Environment Model of Computation
- Loops and Arrays
- Sorting and Searching in Arrays
- Memoization

UNIT 4: BEYOND CONVENTIONAL PROGRAMMING

Learning Objectives

- Stream Processing
- Metacircular Evaluator
- Logic Programming
- Concurrent Programming
- Register Machine

This unit goes beyond conventional high-level programming by covering the paradigms of stream processing, logic programming and concurrent programming. Students will deepen their understanding of computational processes by programming an evaluator for a simple imperative programming language. Finally, a computational model based on register machines connects the students' understanding of computation to the actual computers that carry out computation in practice.



APPLICATION

For more information on the programme and application procedures, please visit https://sicp.comp.nus.edu.sg.

CONTACT

If you have any questions, feel free to send in your enquiries to sicp@comp.nus.edu.sg.

NUS SCHOOL OF COMPUTING

Computing 1
13 Computing Drive
Singapore 117417

https://www.comp.nus.edu.sg/

NUS | Computing
National University of Singapore