L1: Introduction; Elements of Programming

CS1101S: Programming Methodology

Martin Henz

August 11, 2021

The goals of this module

- Understand
  the *structure and interpretation of computer programs*

The goals of this module

- Understand
  the *structure and interpretation of computer programs*
- Learn how to program

The goals of this module

- Understand
  the *structure and interpretation of computer programs*
- Learn how to program
- Get glimpses into the subject areas of computer science

The goals of this module

- Understand
  the *structure and interpretation of computer programs*
- Learn how to program
- Get glimpses into the subject areas of computer science
- Get a feeling for how computer science "ticks"

The goals of this module

- Understand
  the *structure and interpretation of computer programs*
- Learn how to program
- Get glimpses into the subject areas of computer science
- Get a feeling for how computer science "ticks"
- Fall in love

The goals of this module

- Understand
  the *structure and interpretation of computer programs*
- Learn how to program
- Get glimpses into the subject areas of computer science
- Get a feeling for how computer science "ticks"
- Fall in love with computer science

The weekly flow

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts
- Wednesday **P**ath: online post-lecture quiz in Source Academy

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts
- Wednesday **P**ath: online post-lecture quiz in Source Academy
- Thursday 1-h **R**eflection: reflects on lecture material

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts
- Wednesday **P**ath: online post-lecture quiz in Source Academy
- Thursday 1-h **R**eflection: reflects on lecture material
- Friday 1-h **B**rief: special topics

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts
- Wednesday **P**ath: online post-lecture quiz in Source Academy
- Thursday 1-h **R**eflection: reflects on lecture material
- Friday 1-h **B**rief: special topics
- Friday **P**ath: online post-brief quiz in Source Academy

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts
- Wednesday **P**ath: online post-lecture quiz in Source Academy
- Thursday 1-h **R**eflection: reflects on lecture material
- Friday 1-h **B**rief: special topics
- Friday **P**ath: online post-brief quiz in Source Academy
- Monday/Tuesday 2-h **S**tudio session: **Master the concepts!**

The weekly flow

- Wednesday 2-h **L**ecture: introduces new concepts
- Wednesday **P**ath: online post-lecture quiz in Source Academy
- Thursday 1-h **R**eflection: reflects on lecture material
- Friday 1-h **B**rief: special topics
- Friday **P**ath: online post-brief quiz in Source Academy
- Monday/Tuesday 2-h **S**tudio session: **Master the concepts!**
- **M**issions, **Q**uests, **C**ontests: Show your mastery!

Homework to be submitted throughout the semester

Throughout the semester, we will have...

Homework to be submitted throughout the semester

Throughout the semester, we will have...

- 15 Paths

Homework to be submitted throughout the semester

Throughout the semester, we will have...

- 15 Paths
- 20 Missions

Homework to be submitted throughout the semester

Throughout the semester, we will have...

- 15 Paths
- 20 Missions
- 13 Quests

Homework to be submitted throughout the semester

Throughout the semester, we will have...

- 15 Paths
- 20 Missions
- 13 Quests

(some adjustment possible)

This week

This week

- today 2-h **L**ecture L1: Elements of Programming

This week

- today 2-h **L**ecture L1: Elements of Programming
- today **P**ath P1A on Elements of Programming

This week

- today 2-h **L**ecture L1: Elements of Programming
- today **P**ath P1A on Elements of Programming
- tomorrow 1-h **R**eflection R1: choose between 10am and 2pm sessions

This week

- today 2-h **L**ecture L1: Elements of Programming
- today **P**ath P1A on Elements of Programming
- tomorrow 1-h **R**eflection R1: choose between 10am and 2pm sessions
- Friday 1-h Mission **B**rief B1: Runology

This week

- today 2-h **L**ecture L1: Elements of Programming
- today **P**ath P1A on Elements of Programming
- tomorrow 1-h **R**eflection R1: choose between 10am and 2pm sessions
- Friday 1-h Mission **B**rief B1: Runology
- Friday **P**ath P1B on Runology

This week

- today 2-h **L**ecture L1: Elements of Programming
- today **P**ath P1A on Elements of Programming
- tomorrow 1-h **R**eflection R1: choose between 10am and 2pm sessions
- Friday 1-h Mission **B**rief B1: Runology
- Friday **P**ath P1B on Runology
- Friday Mission "Rune Introduction" comes out, due on Tuesday, 17/8

This week

- today 2-h **L**ecture L1: Elements of Programming
- today **P**ath P1A on Elements of Programming
- tomorrow 1-h **R**eflection R1: choose between 10am and 2pm sessions
- Friday 1-h Mission **B**rief B1: Runology
- Friday **P**ath P1B on Runology
- Friday Mission "Rune Introduction" comes out, due on Tuesday, 17/8
- Monday/Tuesday 2-h **S**tudio session S2 on Elements of Programming

Lecturers

Lecturers

Their roles

Conduct **lectures** on Wednesdays and **briefs** on Fridays, coordinate the module

Lecturers

Their roles

Conduct **lectures** on Wednesdays and **briefs** on Fridays, coordinate the module

Martin Henz

Low Kok Lim

Boyd Anderson

Tutors

Their role

Facilitate weekly **Reflection** sessions

Tutors

Their role

Facilitate weekly **Reflection** sessions

Eldric Liew, Kelvin Koa,

Ghozali Suhariyanto Hadi, Hou Ruomu, Liow Jia Chan, Aiden Low,

Martin Henz, Low Kok Lim, Boyd Anderson

Source Academy Team

Their roles

Design, develop, deploy, maintain the **Source Academy**, our learning environment for CS1101S

Avengers

Their role

- facilitating weekly 2-h Studio sessions,
- guiding you in the fine art of programming,
- discussing your missions and quests with you and grading them

Module Overview

- Unit 1—Functions (textbook Chapter 1)
  - Getting acquainted with the elements of programming, using **functional abstraction**
  - Learning to read programs, and using the **substitution model**
  - Example applications: Runes, curves
- Unit 2—Data (textbook Chapter 2)
  - Getting familiar with data: pairs, lists, trees
  - Searching in lists and trees, sorting of lists
  - Example application: sound processing

Module Overview, continued

- Unit 3—State (parts of textbook Chapter 3)
  - Programming with stateful abstractions
  - Arrays, loops, searching in and sorting of arrays
  - Reading programs using the environment model
  - Example applications: robotics, video processing
- Unit 4—Beyond (parts of textbook Chapters 3 and 4)
  - Streams
  - Metalinguistic abstraction
  - Computing with Register Machines / Logic Programming / Concurrency

Assessment timeline

Assessment timeline

- **R**eading **A**ssessment RA1 on Friday Week 4: 3/9 (Unit 1)

Assessment timeline

- **R**eading **A**ssessment RA1 on Friday Week 4: 3/9 (Unit 1)
- **Recess Week**

Assessment timeline

- **R**eading **A**ssessment RA1 on Friday Week 4: 3/9 (Unit 1)
- **Recess Week**
- **Midterm** Assessment on Wednesday Week 7: 29/9 (Unit 2)

Assessment timeline

- **R**eading **A**ssessment RA1 on Friday Week 4: 3/9 (Unit 1)
- **Recess Week**
- **Midterm** Assessment on Wednesday Week 7: 29/9 (Unit 2)
- **R**eading **A**ssessment RA2: 22/10 (Unit 3)

Assessment timeline

- **R**eading **A**ssessment RA1 on Friday Week 4: 3/9 (Unit 1)
- **Recess Week**
- **Midterm** Assessment on Wednesday Week 7: 29/9 (Unit 2)
- **R**eading **A**ssessment RA2: 22/10 (Unit 3)
- **P**ractical **A**ssessment: 10/11 (Unit 4)

Assessment timeline

- **R**eading **A**ssessment RA1 on Friday Week 4: 3/9 (Unit 1)
- **Recess Week**
- **Midterm** Assessment on Wednesday Week 7: 29/9 (Unit 2)
- **R**eading **A**ssessment RA2: 22/10 (Unit 3)
- **P**ractical **A**ssessment: 10/11 (Unit 4)
- **Final** Assessment: 25/11

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...):
0.5% per level, each level has 1000 XP: 36,000 XP

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

3%: Mastery Check 1, any time

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

3%: Mastery Check 1, any time

12%: Midterm Assessment, Week 7

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

3%: Mastery Check 1, any time

12%: Midterm Assessment, Week 7

6%: Reading Assessment 2, Week 10

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

3%: Mastery Check 1, any time

12%: Midterm Assessment, Week 7

6%: Reading Assessment 2, Week 10

12%: Practical Assessment, Week 13

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

3%: Mastery Check 1, any time

12%: Midterm Assessment, Week 7

6%: Reading Assessment 2, Week 10

12%: Practical Assessment, Week 13

3%: Mastery Check 2, any time

Assessment components

18%: Source Academy XP (Missions, Quests, Paths, ...): 0.5% per level, each level has 1000 XP: 36,000 XP

5%: Studio: attendance 3%, effort 2%

5%: Reflection attendance

6%: Reading Assessment 1, Week 4

3%: Mastery Check 1, any time

12%: Midterm Assessment, Week 7

6%: Reading Assessment 2, Week 10

12%: Practical Assessment, Week 13

3%: Mastery Check 2, any time

30%: Final Assessment

Resources

Textbook: SICP JS

Details in LumiNUS > Module Details > SICP JS

Resources

Textbook: SICP JS

Details in LumiNUS > Module Details > SICP JS

Weblinks: Source, Source Academy, Piazza, calendars

LumiNUS > Module Details > Weblinks

Need help?

Need help?

- piazza Q & A forums:
  `piazza.com/nus.edu.sg/fall2021/cs1101s`
  Many of you are signed up, already.
  If you cannot sign up, please email
  `henz@comp.nus.edu.sg`

Need help?

- piazza Q & A forums:
  `piazza.com/nus.edu.sg/fall2021/cs1101s`
  Many of you are signed up, already.
  If you cannot sign up, please email
  `henz@comp.nus.edu.sg`
- PM your Avenger (WhatsApp/FB/Telegram/you-name-it: check what channel they prefer)

Need help?

- piazza Q & A forums:
  `piazza.com/nus.edu.sg/fall2021/cs1101s`
  Many of you are signed up, already.
  If you cannot sign up, please email
  `henz@comp.nus.edu.sg`
- PM your Avenger (WhatsApp/FB/Telegram/you-name-it: check what channel they prefer)
- Email your Reflection instructor:
  LumiNUS > Module Details > Description > Frequently Asked Questions

Need help?

- piazza Q & A forums:
  `piazza.com/nus.edu.sg/fall2021/cs1101s`
  Many of you are signed up, already.
  If you cannot sign up, please email
  `henz@comp.nus.edu.sg`
- PM your Avenger (WhatsApp/FB/Telegram/you-name-it: check what channel they prefer)
- Email your Reflection instructor:
  LumiNUS > Module Details > Description > Frequently Asked Questions
- Email lecturer
  - Boyd: boyd@nus.edu.sg
  - Kok Lim: lowkl@comp.nus.edu
  - Martin: henz@comp.nus.edu.sg

Processes

Processes

Definition (from Wikipedia)

A process is a set of activities that interact to produce a result. The activities unfolds according to patterns that *de*scribe or *pre*scribe the process.

Processes

Definition (from Wikipedia)

A process is a set of activities that interact to produce a result. The activities unfolds according to patterns that *de*scribe or *pre*scribe the process.

Examples

Processes are everywhere. They permeate our nature and culture:

- Galaxies and solar systems follow *formation* processes.
- Metabolic pathways in our bodies are *biological* processes.
- Political parties, legislature, courts, etc. follow processes.
- Industrial production is governed by processes.

Computational processes

Definition

A *computational process* is a set of activities in a computer, designed to achieve a desired result.

Computational processes

Definition

A *computational process* is a set of activities in a computer, designed to achieve a desired result.

Our task

Here we are concerned about how this *design* happens.

Computational processes

Definition

A *computational process* is a set of activities in a computer, designed to achieve a desired result.

Our task

Here we are concerned about how this *design* happens.

Our design method

We use *programs* to prescribe how the computational process unfolds.

What is programming?

What is programming?

People in focus

As the complexity of computer systems increases, *communication* between architects, designers, programmers, operators and users becomes increasingly important.

What is programming?

People in focus

As the complexity of computer systems increases, *communication* between architects, designers, programmers, operators and users becomes increasingly important.

Programs as communication devices

Since programs prescribe the computational processes at the heart of these systems, they can play a central role in the *human* communication during their construction and operation.

What is programming?

People in focus

As the complexity of computer systems increases, *communication* between architects, designers, programmers, operators and users becomes increasingly important.

Programs as communication devices

Since programs prescribe the computational processes at the heart of these systems, they can play a central role in the *human* communication during their construction and operation.

Our motto

Programming is communicating computational processes.

Our programming environment: Source Academy

Our programming environment: Source Academy

Website designed for CS1101S

Our programming environment: Source Academy

Website designed for CS1101S

Has just what you need for understanding the *structure and interpretation of computer programs*.

Our programming environment: Source Academy

Website designed for CS1101S

Has just what you need for understanding the *structure and interpretation of computer programs*.

Research

Source Academy is also a *educational research tool*: We want to to study what happens when people like you learn how to program.

Your first login to Source Academy

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously.

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously. But if you don't want to: No worries: There will be **no penalty** if you don't consent.

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously. But if you don't want to: No worries: There will be **no penalty** if you don't consent.

Our ship

You think you are studying at NUS?

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously. But if you don't want to: No worries: There will be **no penalty** if you don't consent.

Our ship

You think you are studying at NUS? Wrong!

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously. But if you don't want to: No worries: There will be **no penalty** if you don't consent.

Our ship

You think you are studying at NUS? Wrong! Your are cadets in...

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously. But if you don't want to: No worries: There will be **no penalty** if you don't consent.

Our ship

You think you are studying at NUS? Wrong! Your are cadets in... the Source Academy!

Your first login to Source Academy

Consent

You can be part of our research by letting us use your programs anonymously. But if you don't want to: No worries: There will be **no penalty** if you don't consent.

Our ship

You think you are studying at NUS? Wrong! Your are cadets in... the Source Academy!

Visit `https://sourceacademy.nus.edu.sg`, decide on consent and click on "Playground". (Source Academy opens after the lecture.)

Elements of programming

Elements of programming

Every powerful language provides...

Elements of programming

Every powerful language provides...

- Primitives

Elements of programming

Every powerful language provides...

- Primitives
- Combination

Elements of programming

Every powerful language provides...

- Primitives
- Combination
- Means of abstraction

Primitive expressions

Primitive expressions

- Numerals: 0, -42, 486

Primitive expressions

- Numerals: 0, -42, 486
- Numerals use decimal notation

Primitive expressions

- Numerals: 0, -42, 486
- Numerals use decimal notation
- Our interpreter can evaluate numerals, resulting in the numbers they represent

A detail

Expressions are not programs in Source

Instead they can be turned into programs using a semicolon.

A detail

Expressions are not programs in Source

Instead they can be turned into programs using a semicolon.

Example

`486;` is a program.

Means of Combination: Operators

Examples

```
5 * 99;

25 - (4 + 2) * 3;
```

Means of Combination: Operators

Examples

```
5 * 99;

25 - (4 + 2) * 3;
```

Notation as usual

operator between operands: *infix notation* with *precedences*

Evaluating Operator Combinations

But exactly how...

...do we evaluate

```
(2 + 4 * 6) * (3 + 12)
```

???

Evaluation of expressions

Demonstration: Evaluation of expressions

Means of Abstraction: Naming

Example

```
const size = 2;
5 * size;
```

Environment

Purpose

The interpreter of JavaScript keeps track of an environment (table) that associates names with values.

Environment

Purpose

The interpreter of JavaScript keeps track of an environment (table) that associates names with values.

Constant declarations

The keyword `const` adds a name-value entry to the table.

Pre-declared names

Pre-declared constants

Source has a few names pre-declared. For example, the name `math_PI` refers to the constant π.

Pre-declared names

Pre-declared constants

Source has a few names pre-declared. For example, the name `math_PI` refers to the constant π.

Pre-declared functions

In addition to operators such as +, Source has pre-declared functions. For example, `math_sqrt` is the square root function.

Pre-declared names

Pre-declared constants

Source has a few names pre-declared. For example, the name `math_PI` refers to the constant π.

Pre-declared functions

In addition to operators such as +, Source has pre-declared functions. For example, `math_sqrt` is the square root function.

Function application expressions

Functions can be applied as usual in mathematics, for example

```
math_sqrt(15);
```

applies the pre-declared square root function to 15.

Means of Abstraction: Compound functions

Example

```
function square(x) {
    return x * x;
}
```

Means of Abstraction: Compound functions

Example

```
function square(x) {
    return x * x;
}
```

What does this statement mean?

Like constant declarations, this *function declaration* declares a name, here `square`. The value associated with the name is a function that takes an argument `x` and produces (returns) the result of multiplying `x` by itself.

Review: Function application

Recall from the `math_sqrt` example

We apply a function by supplying its arguments in parentheses.

Review: Function application

Recall from the `math_sqrt` example

We apply a function by supplying its arguments in parentheses.

Example

```
function square(x) {
    return x * x;
}
square(14);
```

More examples

```
square(21);

square(2 + 5);

square(square(3));
```

Boolean values

Boolean values

The two boolean values `true` and `false` represent *answers* to yes-no questions

Boolean values

Boolean values

The two boolean values `true` and `false` represent *answers* to yes-no questions

Predicates

A *predicate* is a function or an expression that returns or evaluates to a boolean value.

Boolean values

Boolean values

The two boolean values `true` and `false` represent *answers* to yes-no questions

Predicates

A *predicate* is a function or an expression that returns or evaluates to a boolean value.

Examples

```
true;
false;
x >= 1;
```

Another predicate

Example

```
function adult(x) {
    return x >= 18;
}
```

Conditional expressions

Why?

We would like to check something, e.g. answer a yes/no question, and return a different value, depending on the answer

Conditional expressions

Why?

We would like to check something, e.g. answer a yes/no question, and return a different value, depending on the answer

Back to the example

```
adult(my_age) ? enter_club() : go_to_movie()
```

Conditional expressions

Why?

We would like to check something, e.g. answer a yes/no question, and return a different value, depending on the answer

Back to the example

```
adult(my_age) ? enter_club() : go_to_movie()
```

Example: abs function

To compute the absolute value of a number, check if it is greater or equal 0. If yes, return the number unchanged, and if no, return the number negated.

May the Source be with you!

The Source Acacdemy is open!

May the Source be with you!

The Source Acacdemy is open!

Look out for your first Path "Elements of Programming".

May the Source be with you!

The Source Acacdemy is open!

Look out for your first Path "Elements of Programming".

Welcome on board!