CS1101S Studio Session Week 7:
LEGO Programming, Sorting & Mid-term Review

Niu Yunpeng

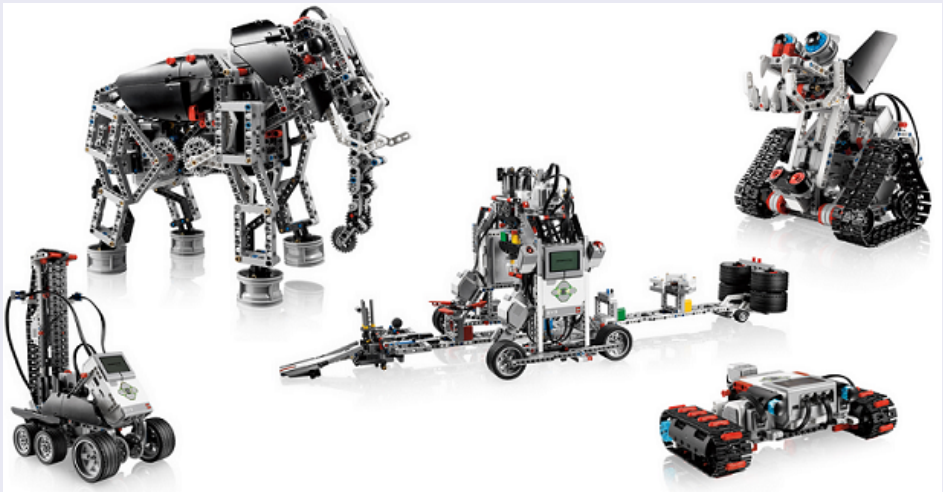
niuyunpeng@u.nus.edu

October 2, 2018

Before We Start

Mission *RoboWarriors* (M7)

- Robot grouping released in Week 6 Studio
- Robot kit issued in Week 7 Studio
- Robot mission assessment in Week 8 Studio
- Robot contest in Week 9 Wednesday night
- Robot kit returned in Week 10 Studio



Mid-Semester Feedback

What do you like about CS1101S Studio?

- A lot of interesting content covered.
- Main concepts in lecture are reviewed and summarized.
- More examples in Studio slides.

What do you dislike about CS1101S Studio?

- Too difficult and tedious.
- Too rushed to cover everything in 2 hours.
- Not enough time to go over every question in Studio Sheet.
- Studio Sheet answer is not provided.
- Hard to follow what the TA is saying sometimes.

Mid-Semester Feedback

Responses from Yunpeng

- Will continue to review concepts and cover more examples in Studio.
- Not enough time: always the problem in CS1101S, let's try to fix it together (don't be late, start on time).
- Studio Sheet answer: forbidden by Prof Martin. Talk to him directly.
- Don't understand what I am saying: ask me to stop and repeat at anytime. Don't be shy.

Overview

1. LEGO programming
   - History of OS and Linux
   - Using ev3dev
   - Robotics programming
2. Sorting
   - Algorithms so far
   - Improvements & more
3. Mid-term review
   - What we have learned
   - To prepare for the mid-term test

Operating system (OS)

Maybe you are familiar with these operating systems:

- Windows
- macOS
- Android
- iOS
- ...




LEGO Programming

Operating system (OS)

But what about them:

- Unix
- Linux
- Ubuntu/Debian/CentOS...

Starting from Unix

- Unix is a pioneer OS that was first developed in 1969 at at the Bell Labs research center by Ken Thompson and Dennis Ritchie, also called *AT&T Unix*.

LEGO Programming

After that...

- Many other OSs have been inspired by Unix philosophy:
  - a set of simple tools (to each perform a limited, well-defined function)
  - a unified file system (as the main means of communication)
  - a shell scripting and command language (to combine the tools to perform complex workfows)
  - modular design.
- These OSs are called Unix-like systems, which is a family of multi-tasking, multi-user computer operating systems.

LEGO Programming

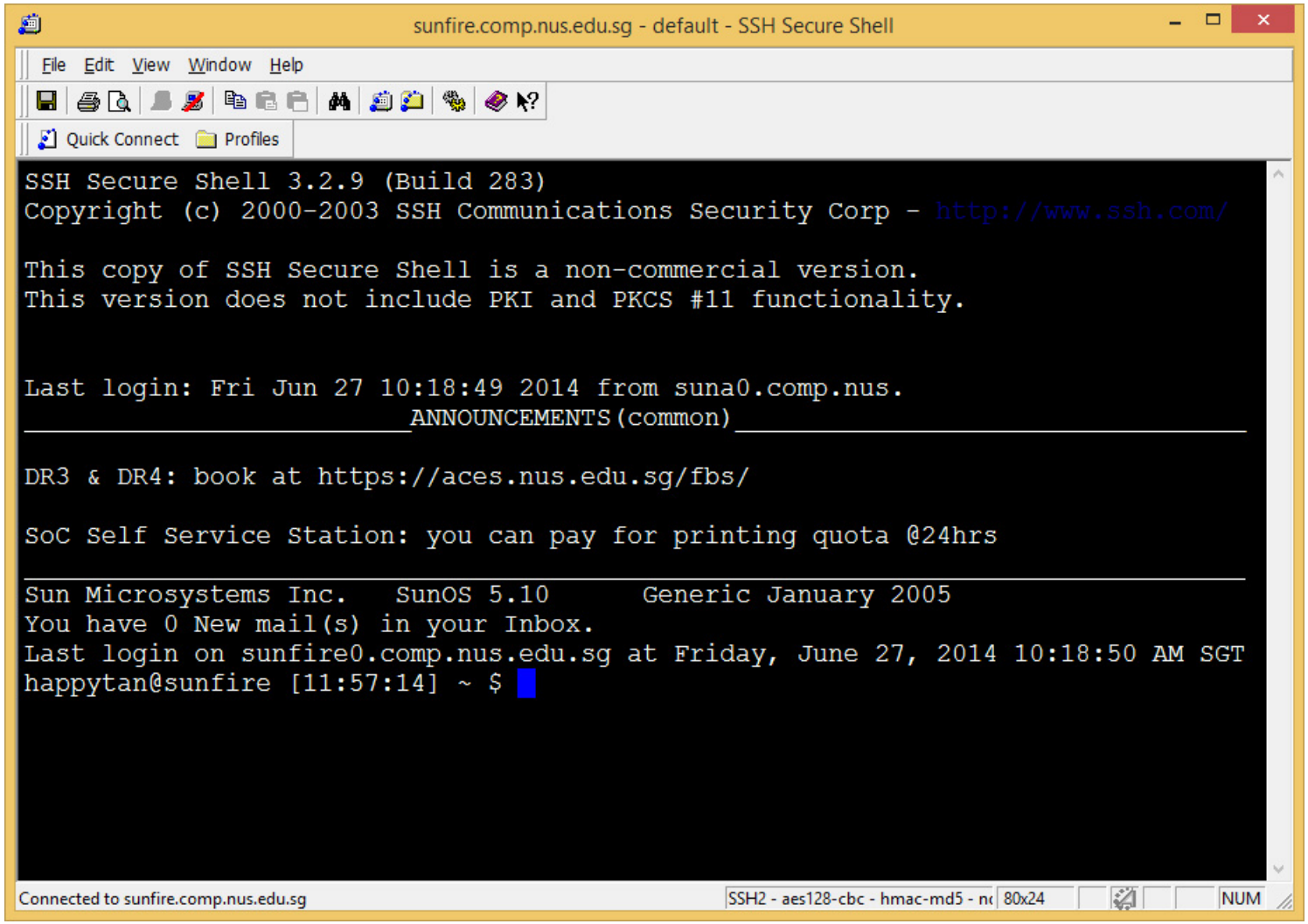
Growing up fast

- Nowadays, Unix-like OS is in fact almost everywhere.
- You may still be not aware that *macOS*, *Linux* and *Android* are all based on *AT&T Unix* and members of the Unix-like family.

Everywhere

- Due to its high performance and reliability, more than 90% of the super-computers around the world are using Unix.
- Our SoC server, **SunFire** is using *Solaris*, a Unix-like OS developed by *Sun Microsystems*.

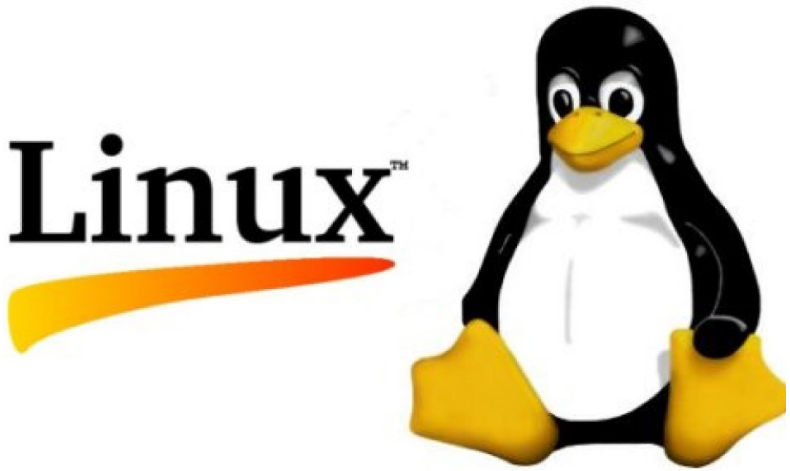
LEGO Programming



LEGO Programming

From Unix to Linux

- Linux was developed by Linus Torvalds in 1991.
- At that time, Linus was still an undergraduate student at University of Helsinki. He was frustrated by the OS used at school then, called Minix. So, he decided to develop a better one by himself.
- If you found any system (like the printers) at SoC very hard to use, you should know why the school makes it to be like that now.



Linux's history

- However, the original Linux should be called *Linux kernel* because it usually performs as a minimum setup instead of full installation.
- Thus, Linux is usually packaged in a form known as *Linux distribution* (or *distro* for short) for both desktop and server usage.
- Some famous *Linux distros* are CentOS, Debian and Ubuntu.

LEGO Programming

Your LEGO ev3 now

- By copying the given image to the SD card, you install ***ev3dev*** for your robot.
- ***ev3dev*** is a variant of *Debian* (a famous Linux distro), which can run on several kinds of LEGO robots.
- Theoretically, you can do any legal Linux operation on ***ev3dev***.

LEGO Programming

To access your ev3dev

- Your ***ev3dev*** is not like your normal laptop OS. It is an embedded system, without monitor, keyboard or mouse.
- However, it does have CPU and memory. So, it can do any task like your normal laptop. But, you need to access it in a different way.

LEGO Programming

To access your ev3dev - use SSH

- SSH is short for *Secure Shell*, a secured method to access from local computer to a remote computer.
- For Windows: use Putty/Pietty/Kitty, OpenSSH, Xshell, etc.
- For mac and Linux: use system built-in Terminal.

Common commands in Linux

- `cd <file_name>`: changes to that selected directory;
- `cd ..`: go back to the parent directory;
- `pwd`: print the absolute path of the current directory;
- `ls`: list all files and sub-directory in the current directory;
  You may want to supply `-a` to include hidden files and `-l` to see the long format (include permission, size, timestamp, etc).
- `rm <file_name>`: remove the selected file;
- `chmod <code> <file_name>`: change the selected file's permission;
- `vim <file_name>`: use vim to edit a file.

Using vim in command-line

- Vim is a simple but powerful text editor in all platforms;
- Vim has two modes: command mode (where you can navigate and manipulate the file, press <ESC> to enter) and insert mode (where you edit the file, press <i> to enter));
- To save and exit: enter command mode, press `:wq<ENTER>`;
- You may want to modify `.vimrc` to change the vim setting (notice that common settings of this file can be found online).

LEGO Programming

Robotics programming

- Robotics programming is exciting because this may be the first time that your program can really make something real move (not on the monitor anymore).
- However, this is not going to be easy. You need to consider more factors.

LEGO Programming

Advice

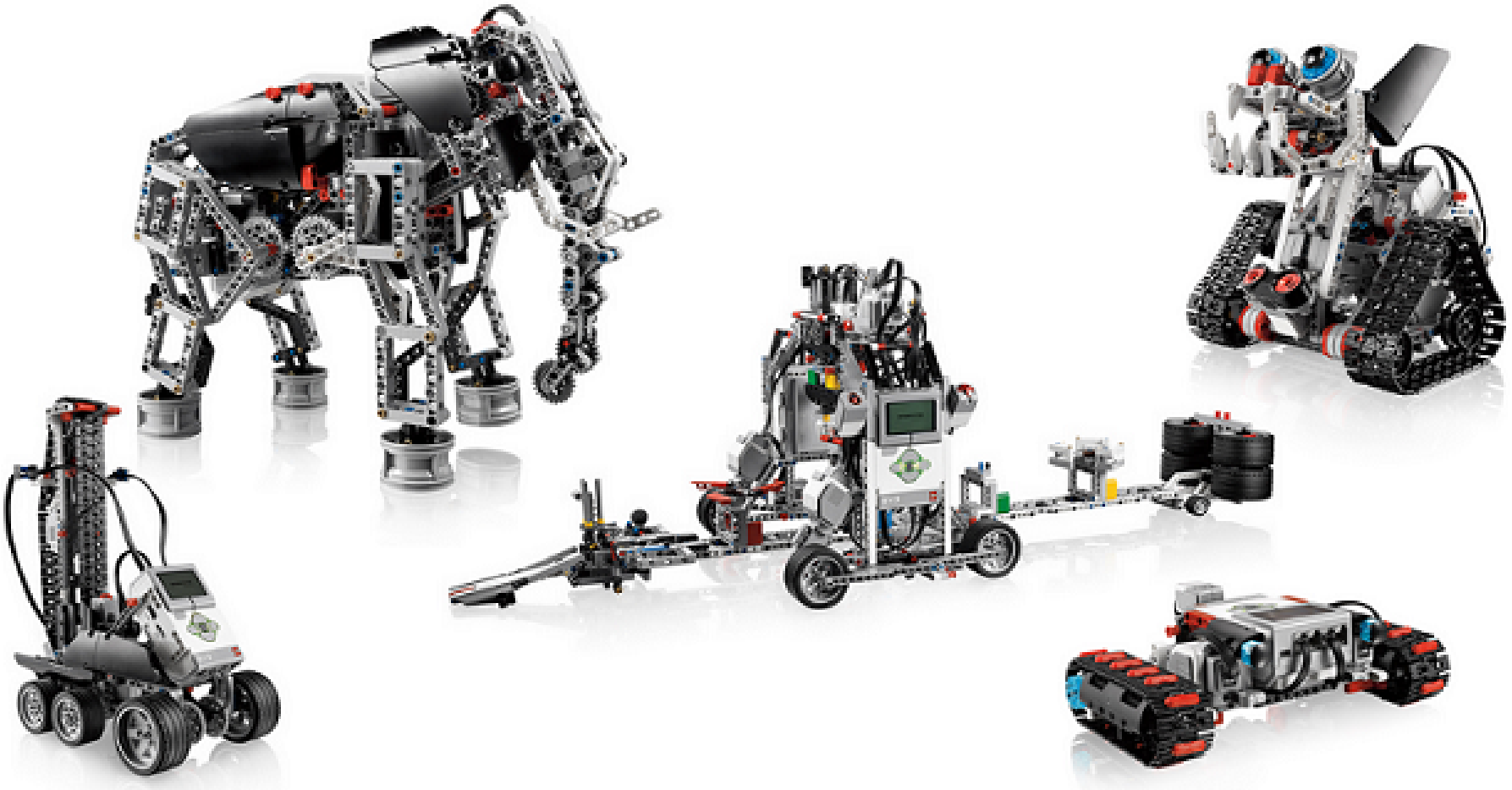
- Remember your math. Try to do some accurate calculation;
- Remember your physics. Gravity, friction, acceleration, ...;
- Remember your programming. Harder to debug this time.

LEGO Programming

A few hints

- Do modular design: each part do independent work;
- Develop your own "callback function": keep doing checks for some conditions, whenever true, the corresponding function will be called;
- The power of the motor may change gradually as you rely on battery.

LEGO Programming



Overview

1. LEGO programming
   - History of OS and Linux
   - Using ev3dev
   - Robotics programming
2. Sorting
   - Algorithms so far
   - Improvements & more
3. Mid-term review
   - What we have learned
   - To prepare for the mid-term test

Sorting

Sorting algorithms so far ...

- Insertion sort
- Selection sort
- Merge sort
- Quick sort

Sorting

Insertion sort

```
function insertion_sort(xs) {
    if (is_empty_list(xs)) {
        return xs;
    } else {
        return insert(head(xs), insertion_sort(tail(xs)));
    }
}
function insert(x, xs) {
    if (is_empty_list(xs)) {
        return list(x);
    } else if (x <= head(xs)) {
        return pair(x, xs);
    } else {
        return pair(head(xs), insert(x, tail(xs)));
    }
}
```

Sorting

Selection sort

```
function selection_sort(xs) {
    if (is_empty_list(xs)) {
        return xs;
    } else {
        const s = smallest(xs);
        return pair(s, selection_sort(remove(s, xs)));
    }
}
```

Sorting

Selection sort

```
function smallest(xs) {
    function sm(x, ys) {
        if (is_empty_list(ys)) {
            return x;
        } else if (x < head(ys)) {
            return sm(x, tail(ys));
        } else {
            return sm(head(ys), tail(ys));
        }
    }

    return sm(head(xs), tail(xs));
}
```

Sorting

Merge sort

```
function merge_sort(xs) {
    if (is_empty_list(xs) || is_empty_list(tail(xs))) {
        return xs;
    } else {
        const mid = middle(length(xs));
        return merge(merge_sort(take(xs, mid)),
                     merge_sort(drop(xs, mid)));
    }
}

function middle(n) {
    return math_floor(n / 2);
}
```

Sorting

Merge sort

```
function merge(xs, ys) {
    if (is_empty_list(xs)) {
        return ys;
    } else if (is_empty_list(ys)) {
        return xs;
    } else {
        const x = head(xs);
        const y = head(ys);

        if (x < y) {
            return pair(x, merge(tail(xs), ys));
        } else {
            return pair(y, merge(xs, tail(ys)));
        }
    }
}
```

Sorting

Merge sort

```
function take(xs, n) {
    if (n === 0) {
        return [];
    } else {
        return pair(head(xs), take(tail(xs), n - 1));
    }
}

function drop(xs, n) {
    if (n === 0) {
        return xs;
    } else {
        return drop(tail(xs), n - 1);
    }
}
```

Sorting

Quick sort

```
function quicksort(xs) {
    // Implementation
}

function partition(xs, p) {
    // Implementation
}
```

More about quicksort ...

- To optimize: an engineering task
  - Use dual-pivot quicksort (implemented in Java 8)
  - Switch to insertion sort when divided until size is small, say < 1000.
- To avoid the worst case: select the pivot smartly
  - Select a random pivot?
  - Use paranoid quicksort?
- To save space: use in-place partition routine
- To be stable (when there are duplicates): 3-way partition
  - Two pass? One pass?
- What if the size of dataset is too large?
  - We cannot even load all data required to be sorted into memory.
  - Use **external sorting**!

Sorting

More sorting algorithms in the future ...

- Comparison-based sorting
  - Bubble sort, heap sort, ...
- Non-comparison-based sorting
  - Radix sort, counting sort, bucket sort, ...

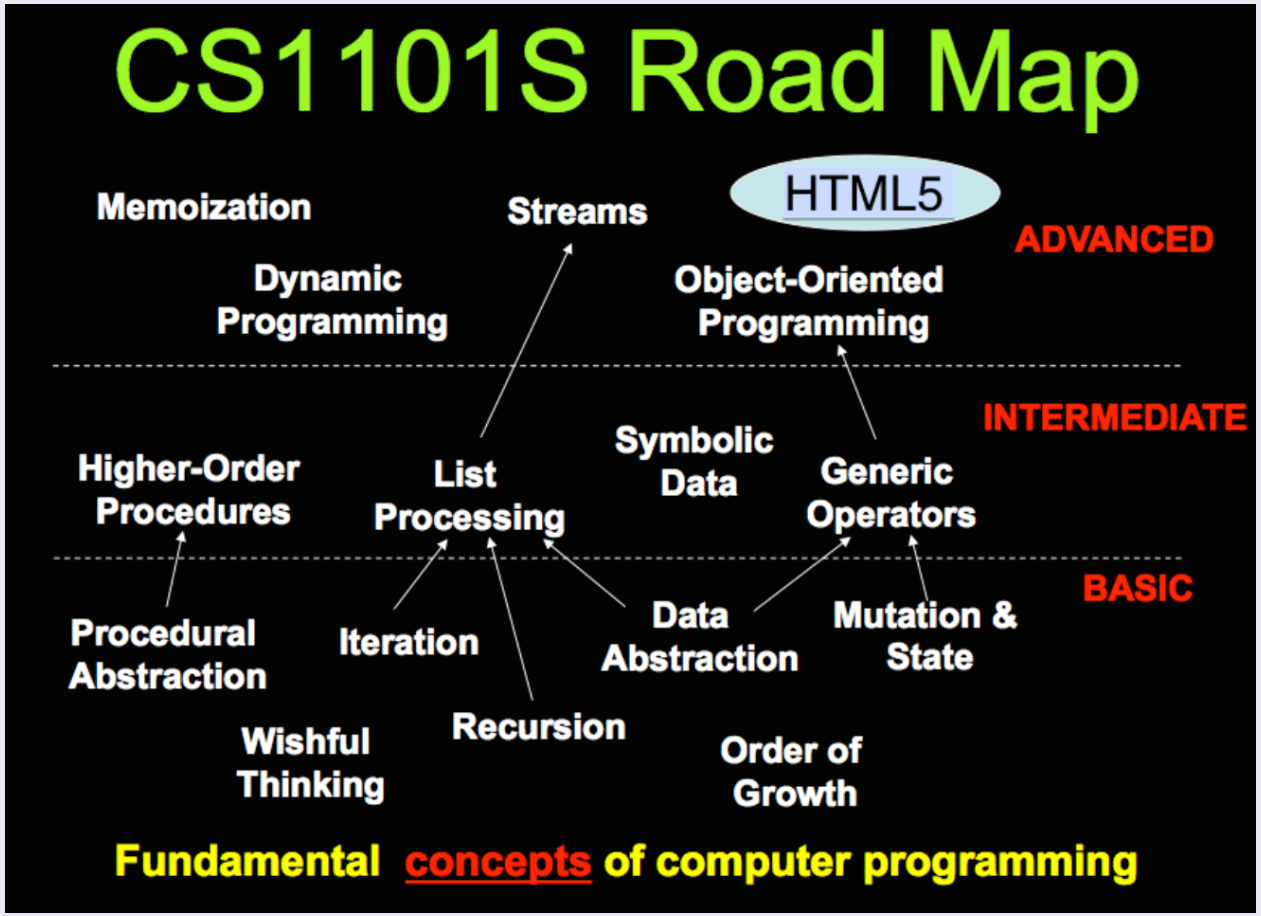
Lower bound ...

- For comparison-based sorting: $\Omega(n \cdot logn)$

Overview

1. LEGO programming
   - History of OS and Linux
   - Using ev3dev
   - Robotics programming
2. Sorting
   - Algorithms so far
   - Improvements & more
3. Mid-term review
   - What we have learned
   - To prepare for the mid-term test

Revisit the CS1101S roadmap



Mid-term Review

Things we have covered so far...

- Components of programming language
- Wishful thinking/abstraction
- Recursion/iteration
- Higher-order programming
- Pair/list/tree processing
- Data structure design
- ...

Mid-term Review

Components of programming language

- Primitives:
  The smallest constituent unit of a programming language.
- Combination:
  Ways to put primitives together.
- Abstraction:
  The method to simplify the messy combinations.
  - To abstract data: use naming;
  - To abstract procedures: use functions.
  - Sometimes, naming and functions are combined together.

Wishful thinking/abstraction

To make a good abstraction:

- Modularity:
  Separate multiple steps (and sub-steps).
- Readability:
  Easy for others to read and understand.
- Reusability:
  Provide a generic interface to be used commonly.
- Maintainability:
  Convenient to debug, refactor and deploy.

Mid-term Review

Recursion/iteration

- *Iteration*: the buttom-up approach;
- *Recursion*: the top-down approach.

How to understand recursion?

- Use ***substitution model***.
- Repeatedly replace a function call by its function body, in which the formal parameters are replaced by the respective actual arguments.

Mid-term Review

Recursive function

- Any function that calls itself (directly or indirectly) is called a recursive function.

To write recursive functions correctly

- Base case(s)
- Scale
- Sub-problem(s)

Mid-term Review

Deferred operation

- The operations that have to be suspended because they need to wait for some other operations to finish first.
- In order to suspend them, we need to remember them in the memory, which is a waste of space.

Recursive & iterative process

- Execution of a recursive function may give rise to either a recursive or iterative process.
- Recursive process: those with deferred operations.
- Iterative process: those without deferred operations.

Mid-term Review

Classical examples of recursion

- Factorial
- Square root
- Power function
- Fibonacci
- Greatest common divisor (GCD)
- Least common multiple (LCM)
- Hanoi tower
- Coin change
- Permutation / combination
- ...

Higher-order programming

Why we can do higher-order programming:

- Functions are also variables.
- They are not special.
- They just behave like normal variables.

To use higher-order programming:

- Constants can be functions.
- Parameters can be functions.
- Return values can be functions.

Pair/list/tree processing

Up to now, the list library supports different kinds of functions:

- List builder: `list`, `build_list`, `enum_list`;
- List getter: `head`, `tail`, `list_ref`, `member`, `is_member`;
- List information: `is_list`, `is_empty_list`, `length`, `equal`;
- List modifier: `append`, `reverse`, `remove`, `remove_all`, `filter`, `map`, `for_each`;
- List converter: `accumulate`, `list_to_string`.

Mid-term Review

Data structure design

You should follow these principles:

- Understand the requirement before doing the actual design;
- Separate the interface from the implementation;
- Compare the advantage and tradeoff;
- Principle of last commitment.

Mid-term Review

Two types of study

- Subject-oriented: to learn the really useful stuff;
- Examination-oriented: to help you get good grades.

Consequence

- Subject-oriented: good for you (long-term goal);
- Examination-oriented: good for your CAP (short-term goal).

Mid-term Review

How to choose between two types of study

- During recess week and reading week: *examination-oriented*;
- Else: *subject-oriented*.

Suggestion

- CAP is important that it should be part of your life.
- However, it should not become all of your life.

Mid-term Review

To prepare for an examination effectively

- Read all the materials again;
- Do as many PYPs (past year papers) as possible;
- Summarize what you have learned;
- Be relaxed.

Mid-term Review

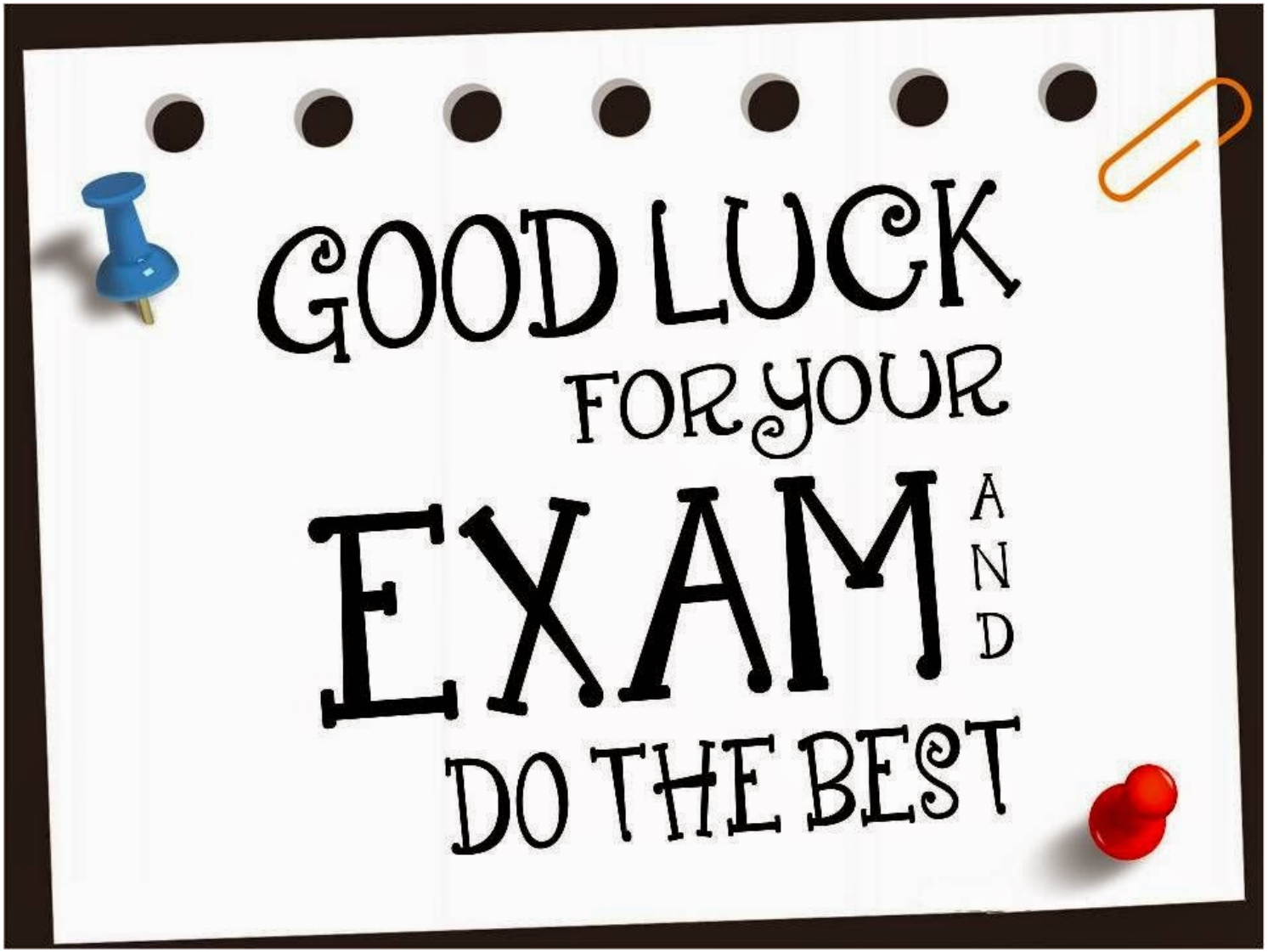
To prepare for CS1101S mid-term test

- Do all the available PYPs carefully;
- Read all lecture notes, recitation notes, studio notes again;
- Do all studio group problems again;
- Be familiar with the latest Source language library;
- If you still have time, read the textbook *SICP*.

After these steps ...

- Don't worry anymore, you are ready for the midterm!

Good Luck



The End

Copyright