COMP10002

Foundations of Algorithms

Algorithms are fun!

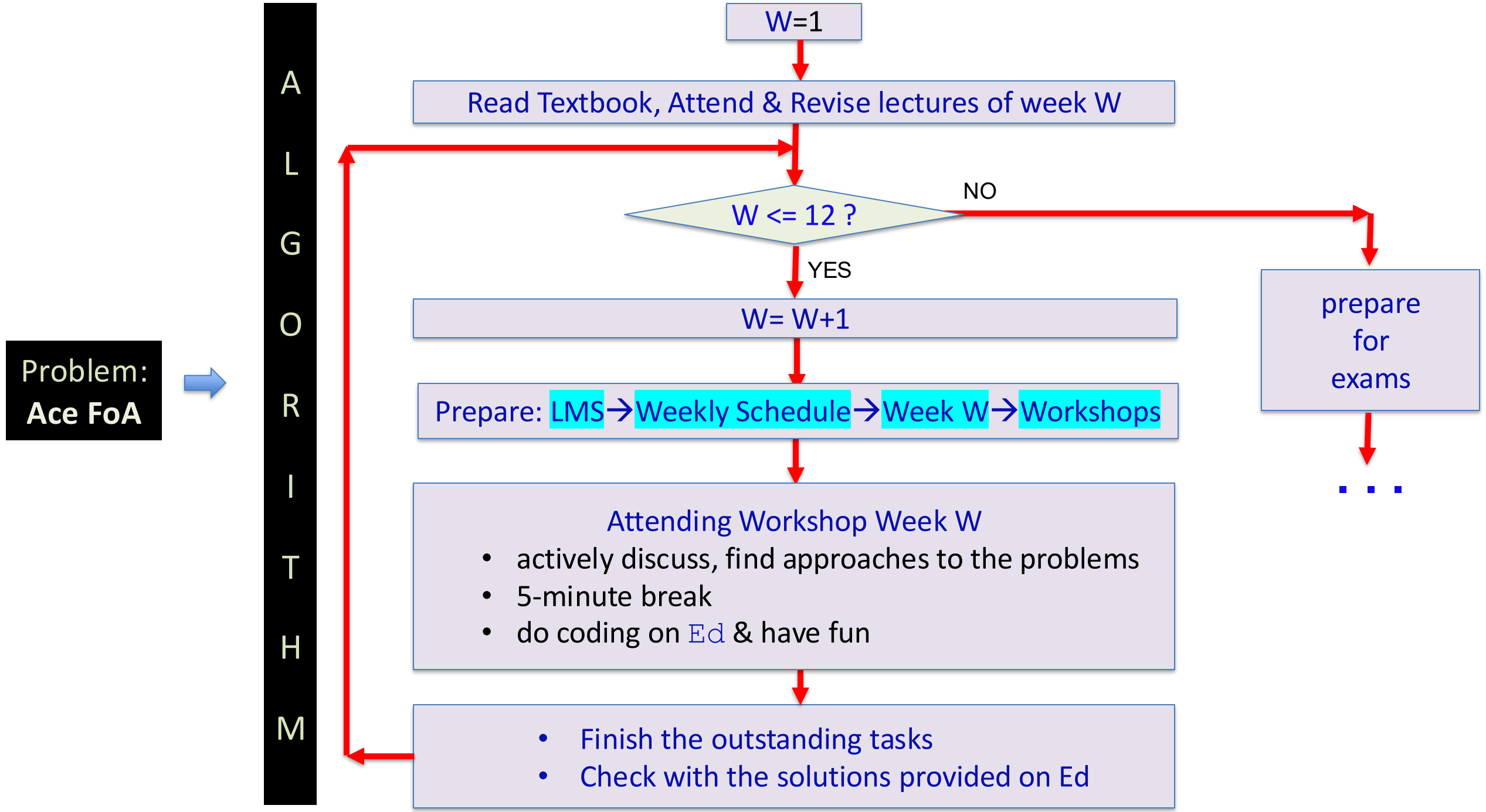
Welcome to the First Workshop!

When waiting

- greet and know friends around you, and
- open **LMS** and **Ed** in your laptop (or a lab's PC)

Today's Plan:

- About Us & the Workshops
- Discussions:
  - Problems → Algorithms → Programs, Python vs C
  - Our first computational problems: 2.08, 3.07
- 5-min break
- Lab: using `Ed` for exercises 1.02, 2.08, 3.07 and (hopefully) more


Problem:
Ace FoA
ALGORITHM
W=1
Read Textbook, Attend & Revise lectures of week W
W <= 12 ?
NO
YES
prepare for exams
W= W+1
Prepare: LMS→Weekly Schedule→Week W→Workshops
Attending Workshop Week W
• actively discuss, find approaches to the problems
• 5-minute break
• do coding on Ed & have fun
• Finish the outstanding tasks
• Check with the solutions provided on Ed

Our Working Platform: Ed

Use Discussion Forum on Ed:

- don’t hesitate to post questions,
- remember: there is no silly question, also you can post anonymously
- answer questions if you can

Use Ed for programming during the workshop and anywhere:

- It’s a powerful online programming platform
- Weekly exercises and solutions are there
- Learn to use Ed effectively using a guide in LMS:

  LMS → Modules , then a long scrolling down

  → Compiling and Editing: Guides to Using Ed Lessons

Try now with Exercise 1.02 (helloworld.c) on Ed

Discussion 1: PAP = Problems ➔ Algorithms ➔ Programs

Solving a computational problem in computers:

1. **Understanding the problem:** Clearly defining inputs and outputs helps in this phase. Questions:
    - input: how many inputs, what's the data type of each input
    - output: how many outputs, what's the data type of each output
    - how to produce output from input: ideas/approaches to produce output from input)

2. **Building an algorithm:** step-by-step procedure to produce output from input

3. **Coding, testing, and debugging:**
    - Translate the algorithm into (Python or, why not?, C or....) code,
    - verify its correctness, and
    - fix all issues.

4. **Practical using:** Deploy the solution and potentially refine it based on user feedback.

Examples of the first 2 stages

Problem 2.08: convert Fahrenheit degrees to Celsius degrees

Input:

Output:

Algorithm:

Problem 3.07: add the reverse conversion (from Celsius to Fahrenheit) to 2.08

Input:

Output:

Algorithm:

Discussion 2: Why C? C vs Python (as we have seen)

Python	C

# no declarations needed 	
# no type 	
a= 10
b= 4
# we can change a 	
# to a string like "ABBA" if we like! 	
	int a, b; // variables must be declared before use
	// variables are *typed*
	a= 10; // statement must end with ;
	b= 4;
	// a and b are int and remain to be int till the end

if (a>b) :
print("max= ", a)
print("a/b= ", a/b)
else :
max= b
 	
# code block defined by indentation 	
	if (a>b) {
	printf("max= %d\n", a);
	printf("a/b= %d\n", a/b);
	} else {
	x= b;
	}
	// Code block enclosed in { ... }
	// Indentations not required but desirable

How about assignment and expressions in C vs in Python?

output: using a format

Output using: `printf`: print according to a format

Python	C

a= 10
b= 4
if (a>b) :
  print(“max= ”, a)
  print(“a/b= “, a/b)
else :
  max= b
  . . .
	int a, b;
	a= 10;
	b= 4;
	if (a>b) {
	  printf(“max= %d\n”, a);
	  printf(“a/b= %d\n”, a/b);
	} else {
	  x= b;
	  . . .
	}

Outputs	
max= 10 a/b= 2.5	max= 10 a/b= 2

input with scanf: also uses a format

scanf returns the number of data it successfully reads.

the program continues even if scanf fails to read some values

Python	C

`a= input("a, b = ")`
`b= input()`
`print (a, b)`
	`int a= 0, b= 0, n= 0;`
	`printf("a, b= ");`
	`n= scanf("%d%d", &a, &b);`
	`printf("C: n= %d, a= %d, b=%d\n", n,a,b);`

Input: 10 ⏎ 20 ⏎ **10 20**	**C: n= 2, a= 10, b= 20**
Input: 10 20 ⏎ (stops, waiting)	**C: n= 2, a= 10, b= 20**
Input: 10.20 ⏎ (stops, waiting)	**C: n= 1, a= 10, b= 0** (not as expected, but continue working) (next scanf reads from .20)

when using `scanf`, we should check for successfulness

✗	✓

`scanf("%d", &m);`
 	
`//` not good: `m` may not receive value
	`if (scanf("%d", &m) != 1) {`
	`  printf("Invalid data\n");`
	`  exit(EXIT_FAILURE);`
	`}`
	`//` all good for continuing

Discuss with your classmates to see if you get the same answer

Suppose that we have declarations:

```
double x; char c;
```

and the intended input data (from keyboard) looks like

```
100A
```

What is the best code fragment for reading value for `x` *and* `c` *(to get* `100` *for* `x`*,* `'A'` *for* `c`*)?*

A	`if (scanf("%lf%c", &x, &c) == EOF){`
	`  exit(EXIT_FAILURE);`
	`}`

B	`if (scanf("%lf %c", &x, &c) != 2){`
	`  exit(EXIT_FAILURE);`
	`}`

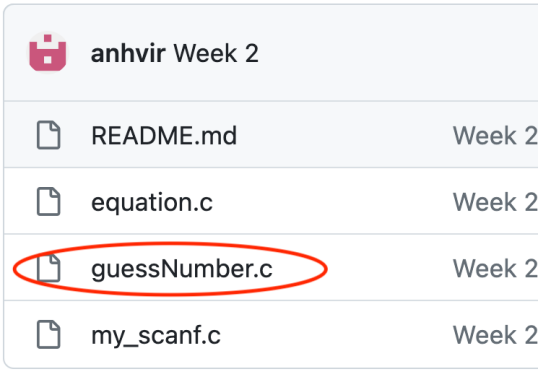
C	`scanf("%lf%c",&x, &c);`

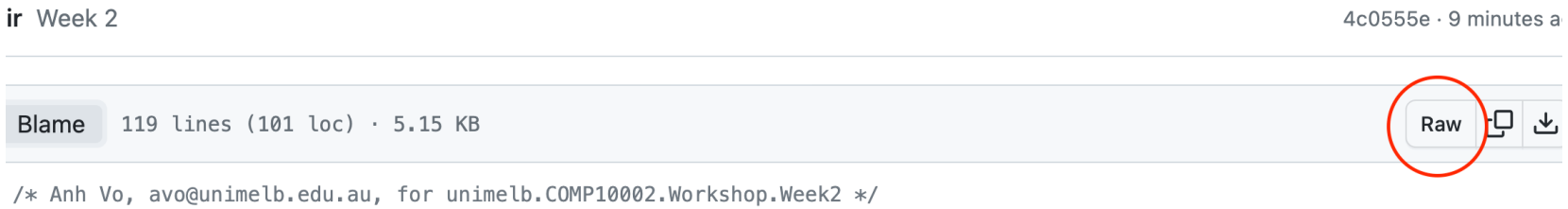
D	`if (scanf("%f%c", &x, &c) == 2) {`
	`  exit(EXIT_FAILURE);`
	`}`

E	all the above options are incorrect

5-minute break: Social Networking + [optionally] having fun with:

in a new windows of your browser, go to github.com/anhvir/c102
Click on guessNumber.c → Raw Press Ctrl-A then Ctrl-C to copy the Raw content (to clipboard)
(Or just try the simpler and more interesting hello.c)





Switch to Ed:

1. In an empty exercise (e.g., Ex 1.02), click Mark. This also stores the current program.c as your latest submission, ensuring that the content is safely preserved.
2. Paste the new code (the content of clipboard) into program.c.
3. Click Run to try it, then explore the code.
4. Want to get back your previous program.c? Click  n the top)

Lab Time == Fun Time ☺

Minimal

Exercise	Content and Notes
`1.02`	`Hello World`
`2.08`	convert F -> C
`3.07`	extend `2.08` by adding in the reverse transformation

And also (why not?)

`3.07b`	extend **`3.07`** by adding conversions for lengths and masses.
`3.04`	print out the date of tomorrow (excellent real-life problem, you will use heap of `if`)
	other exercises from `Chapter 03` and `Chapter 02`

- Share the fun and discuss with your classmates!
- Raise your hand if getting stuck or getting excited.

- When doing a `Ed` exercise at home, you can post questions on Ed discussion forum, and a tutor or Alistair/Artem will answer.
- Solution for each exercise will be available on Ed a week after.

Wrap-Up

Algorithms & Programming are fun

Get help from:

- LMS, Lectures, Textbook, other books
- discussion forum
- classmates
- CIS First Year Centre (FYC)
- emailing Anh: avo@unimelb.edu.au
- responsible & wise use of Gemini, ChatGPT, Google, ...

FYC

- in Level 3, Melbourne Connect
- opens every workday, 2 PM - 4 PM
- has a tutor waiting to answer questions!

Remember:

	`int`	`float`	`double`	`char`	*string*
`printf` format	`%d`	`%f`	`%lf`	`%c`	`%s`
`scanf` format	`%d`	`%f`	`%lf`	`%c`	
`scanf` for **`v`**	`&v`	`&v`	`&v`	`&v`	

Expressions: Unlike **Python**, in **C**:

- int/int → int (truncated)
- assignment a= exp also is an expression and has value exp
- logical like a&&b has int value: 1 (TRUE) or 0 (FALSE)
- similarly, comparison like a<b has value 1 or 0

```
if (guard) {
   ...
} else {
   ...
}
// guard: TRUE if being non-zero
//        FALSE otherwise
// but there is no Boolean type
```