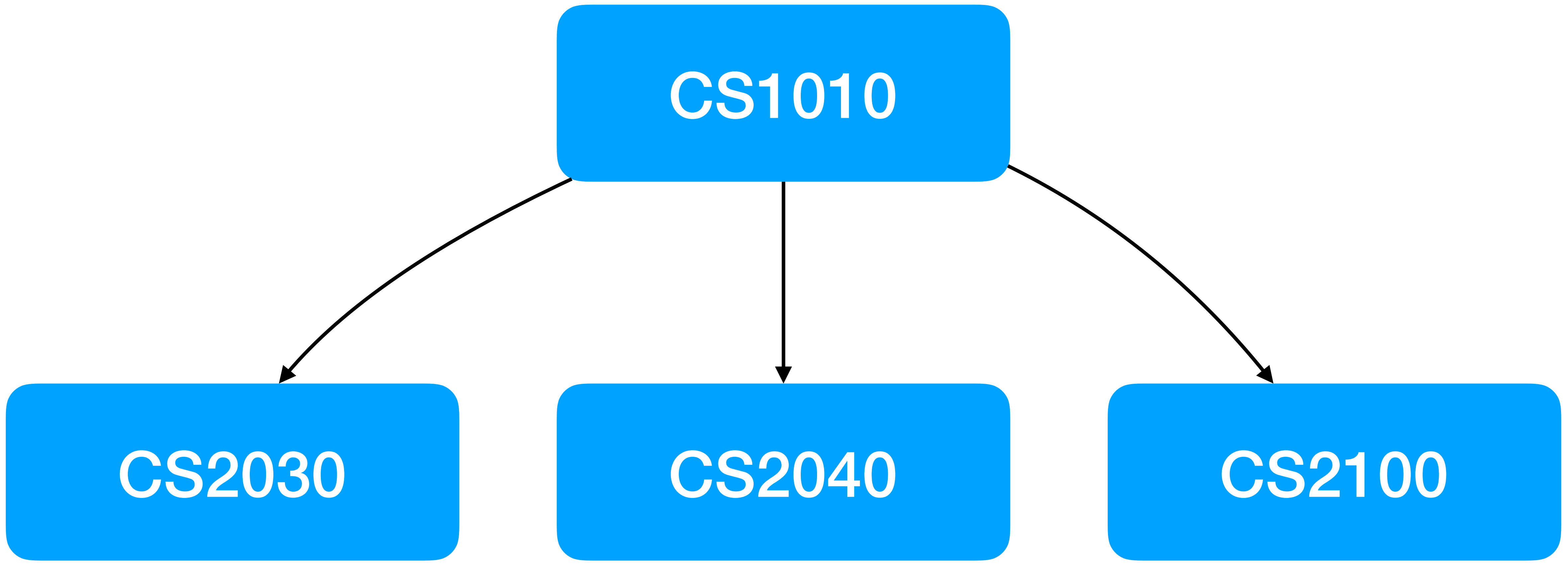
CS2030

Programming Methodologies II

Ooi **Wei Tsang**

ooiwt@comp.nus.edu.sg
AS6 05-15

CS2030 Office Hours:
Monday 2-3pm



Software Principles and Engineering

Theory and Algorithms

Computing Systems

What should you know?

- Basic programming methodology (functions, loops, conditionals, recursion, debugging, testing, etc)
- Ability to learn new programming language syntax quickly
- Comfortable in writing small programs (order of hundreds of lines)

What will you learn?

- Understand and apply advanced programming concepts, including object-oriented concepts and functional programming concepts and constructs
- Comfortable with reading and developing medium-scale software (thousands of lines)
- Become proficient with modern Java (1.8 or above)

- know the pros/cons of OOP and FP, and when to use which
- be able to pick up new languages and advanced programming concepts quickly

Important Dates

March 5, 2018 (Midterm)

May 3, 2018 (Final)

Open Book Exams

Not a memorisation-based module

Any APIs needed will be given

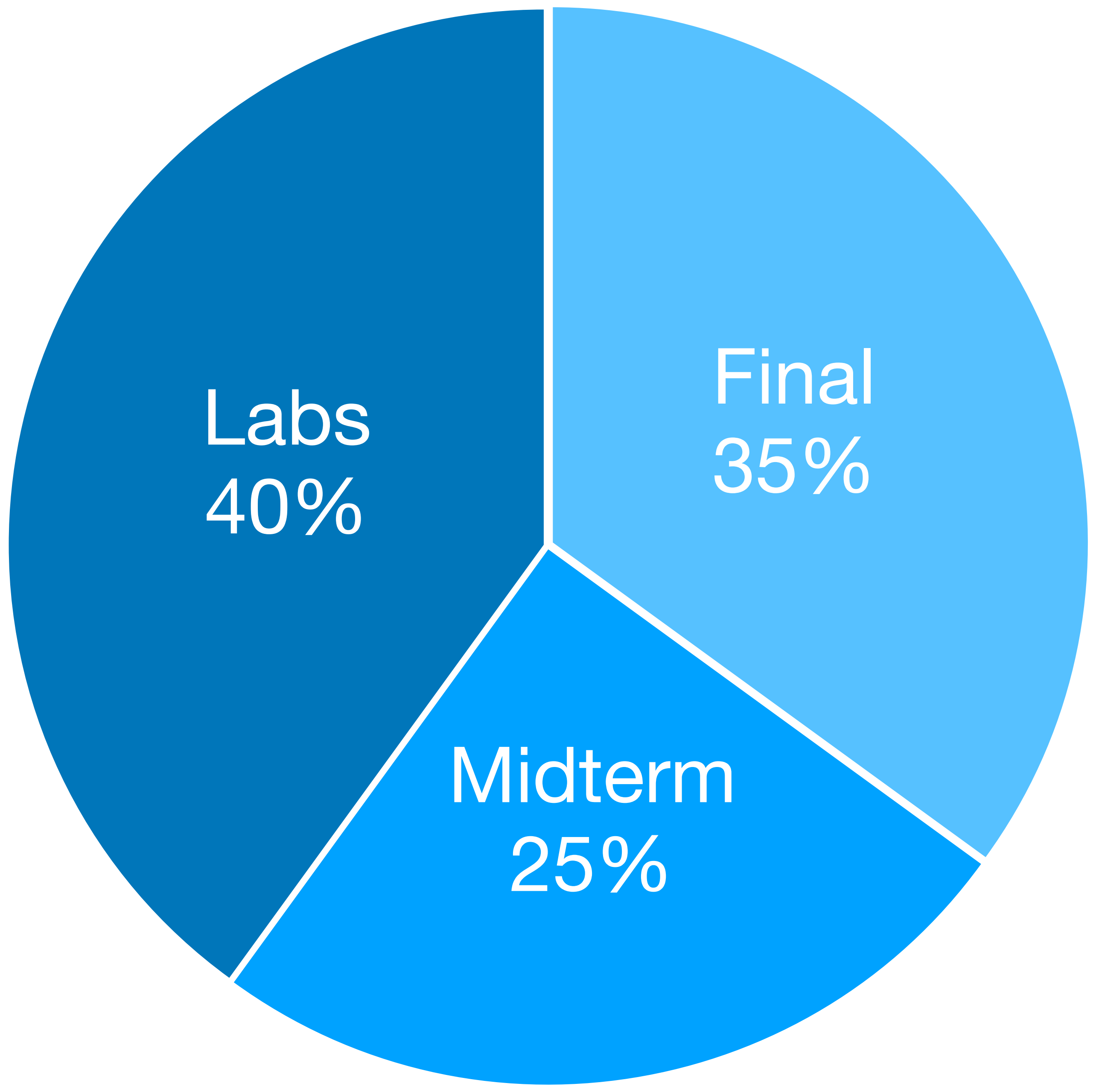
Module Homepage

`https://nus-cs2030.github.io/1718-s2/`

Piazza Q&A

https://piazza.com/class/jcaaskvbs754wh#

Assessment



Labs

- Where the learning happens
- Mixed of graded and ungraded labs
- Graded labs built on top of ungraded labs
- Two hours lab session per week (attendance is highly encouraged)

Strict policy on plagiarism

Disciplinary action will be taken :(

TODO

- Register for an SoC UNIX account if you do not have one at https://mysoc.nus.edu.sg/~newacct/
- Read about the class policy at https://nus-cs2030.github.io/1718-s2/policies/index.html
- Activate your Piazza account (link emailed to you)
- Download and install Java SE 9 (aka JDK 9) on your laptop https://nus-cs2030.github.io/1718-s2/jdk/index.html

Light a Fire

Flickr photo by danielygo Some rights reserved

Not Fill a Bucket

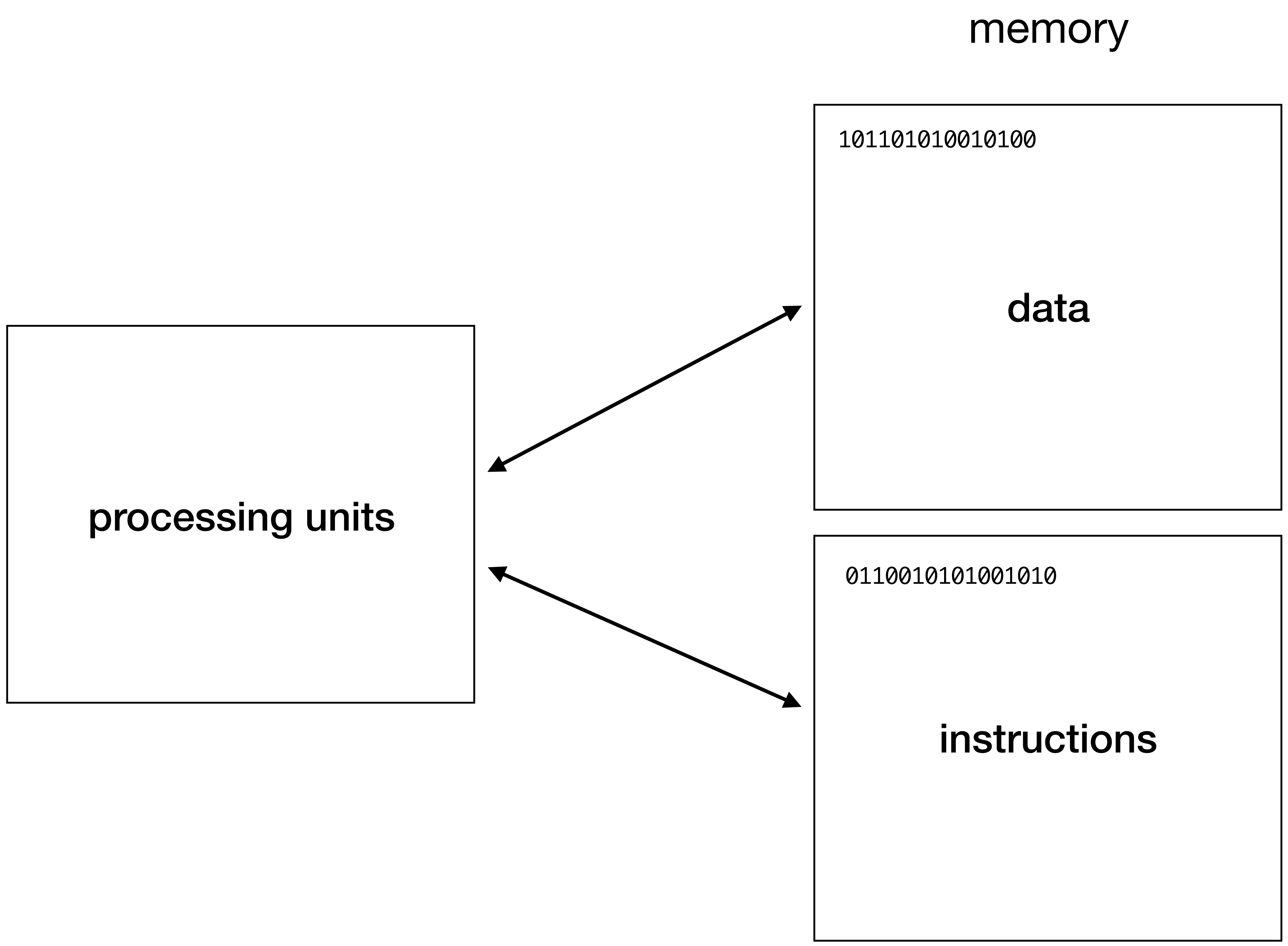
Flickr photo by peasap Some rights reserved

Lecture 1

Abstraction and Encapsulation

What is a program?

Very very high-level view



What is a type?

0100 0001

`0100 0001 ..`

character integer memory address

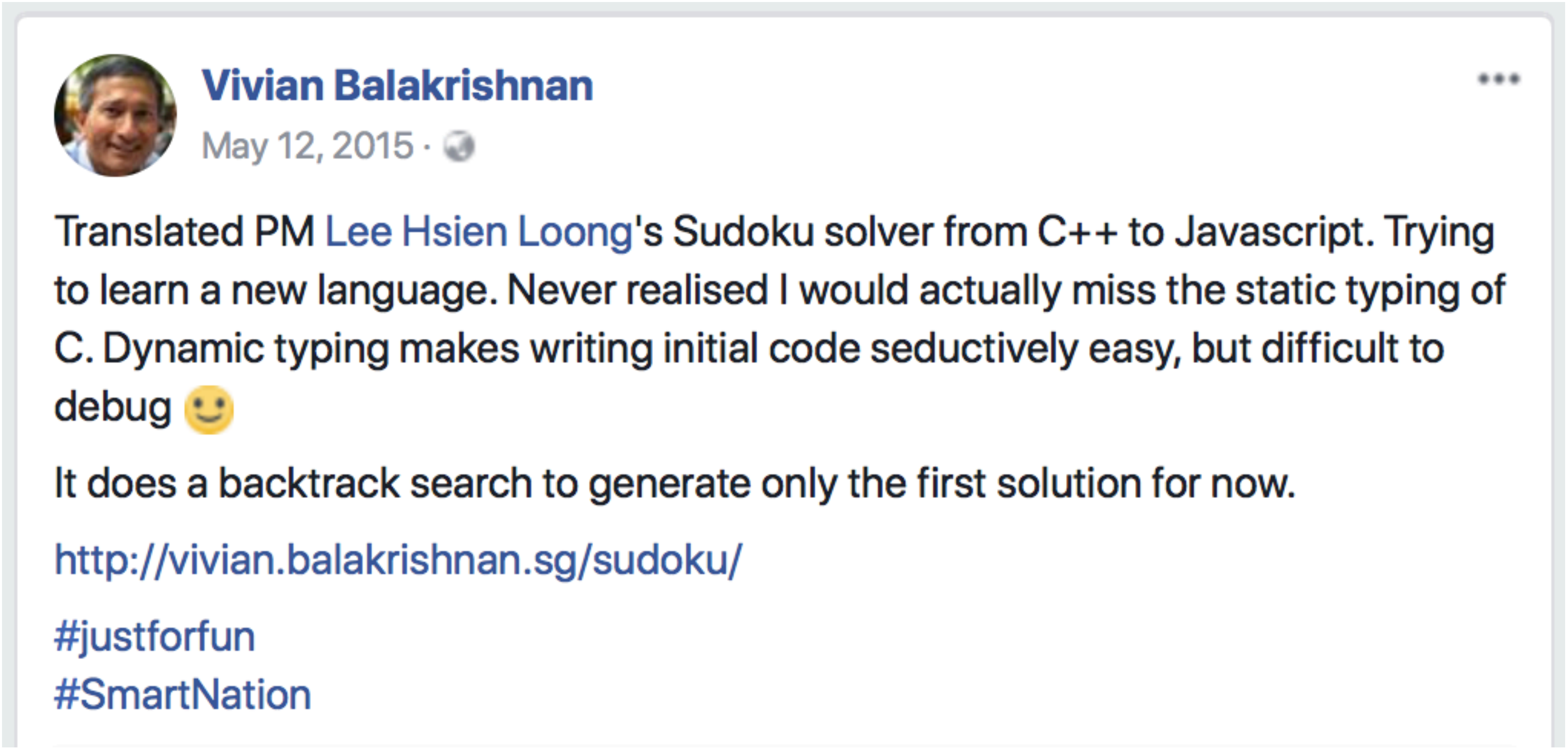
Dynamic vs. **Static Type**

Static type

```
int x;
x = 3;
x = “hello”; // error
```

Dynamic type

```
x = 3
x = “hello” // ok
```



Vivian Balakrishnan

May 12, 2015 ·

Translated PM Lee Hsien Loong's Sudoku solver from C++ to Javascript. Trying to learn a new language. Never realised I would actually miss the static typing of C. Dynamic typing makes writing initial code seductively easy, but difficult to debug 🙂

It does a backtrack search to generate only the first solution for now.

http://vivian.balakrishnan.sg/sudoku/

#justforfun
#SmartNation

Abstraction: Function

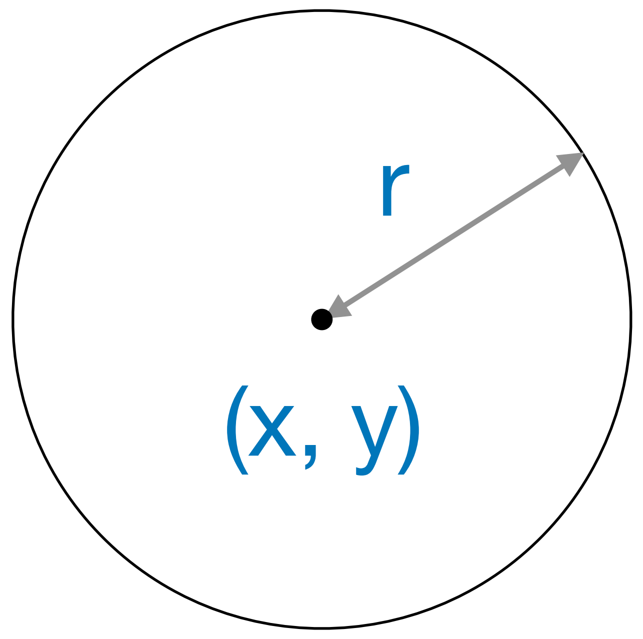
```
int i, k, j;
for (i = 1; i < n; i++)
{
    k = a[i];
    j = i-1;
    while (j >= 0 && a[j] > k)
    {
        a[j+1] = a[j];
        j = j-1;
    }
    a[j+1] = k;
}
```

$\longrightarrow$

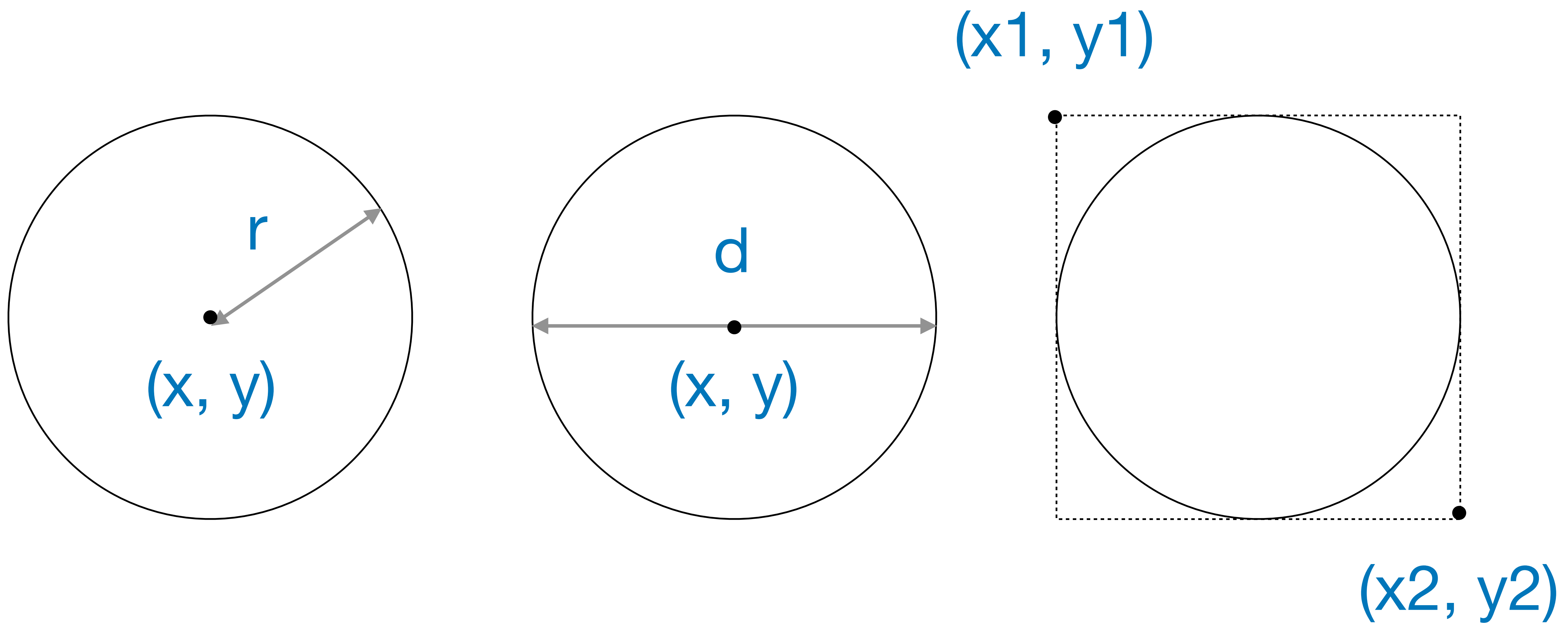
```
sort(a)
```

Abstraction: Composite Data Type

```
struct circle {
  double x;
  double y;
  double r;
}
```

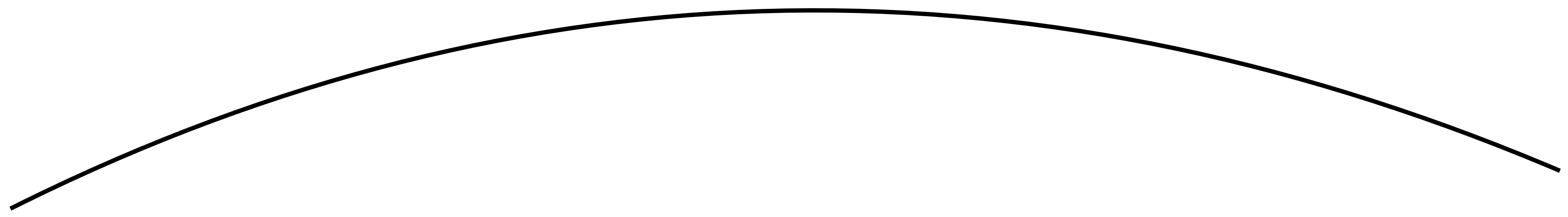


```
area = 3.1415 * r * r;
```

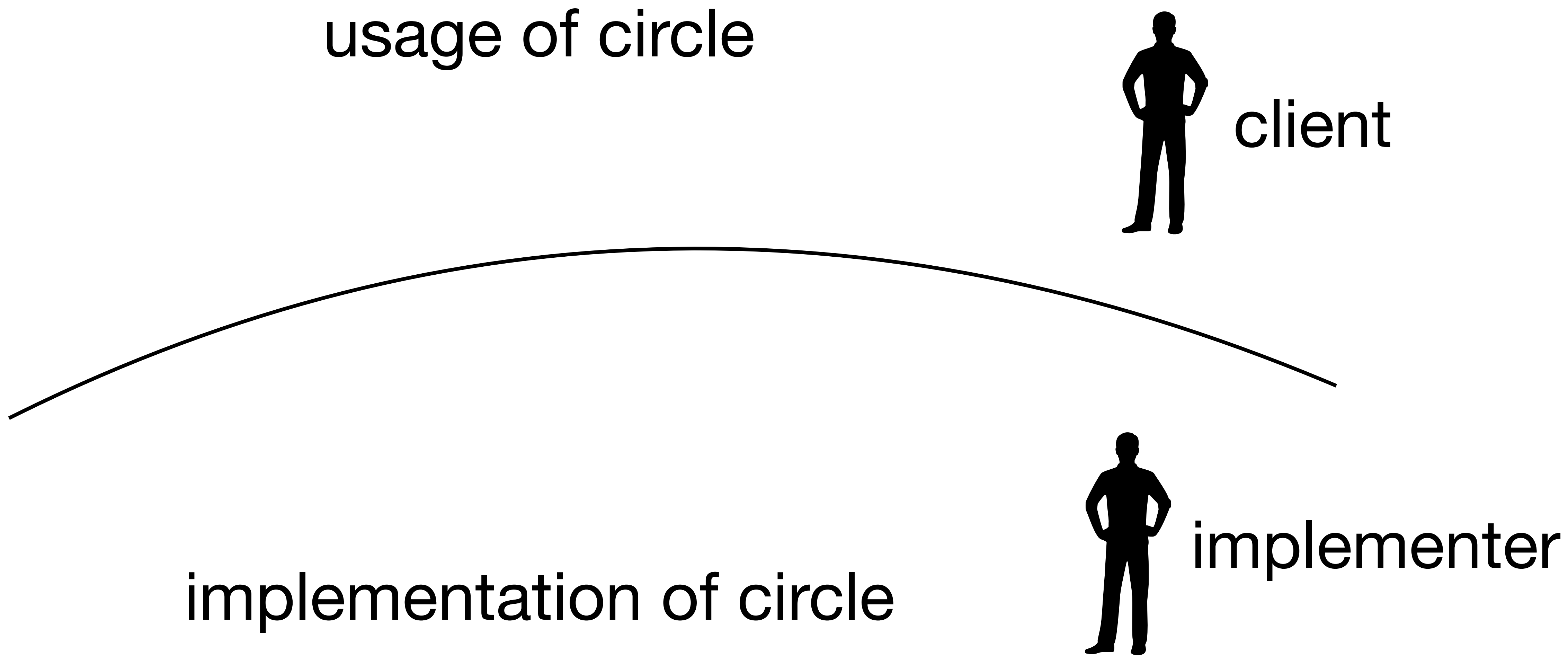


```
area = 3.1416*r*r;     area = 0.7854*d*d;    area = …
```

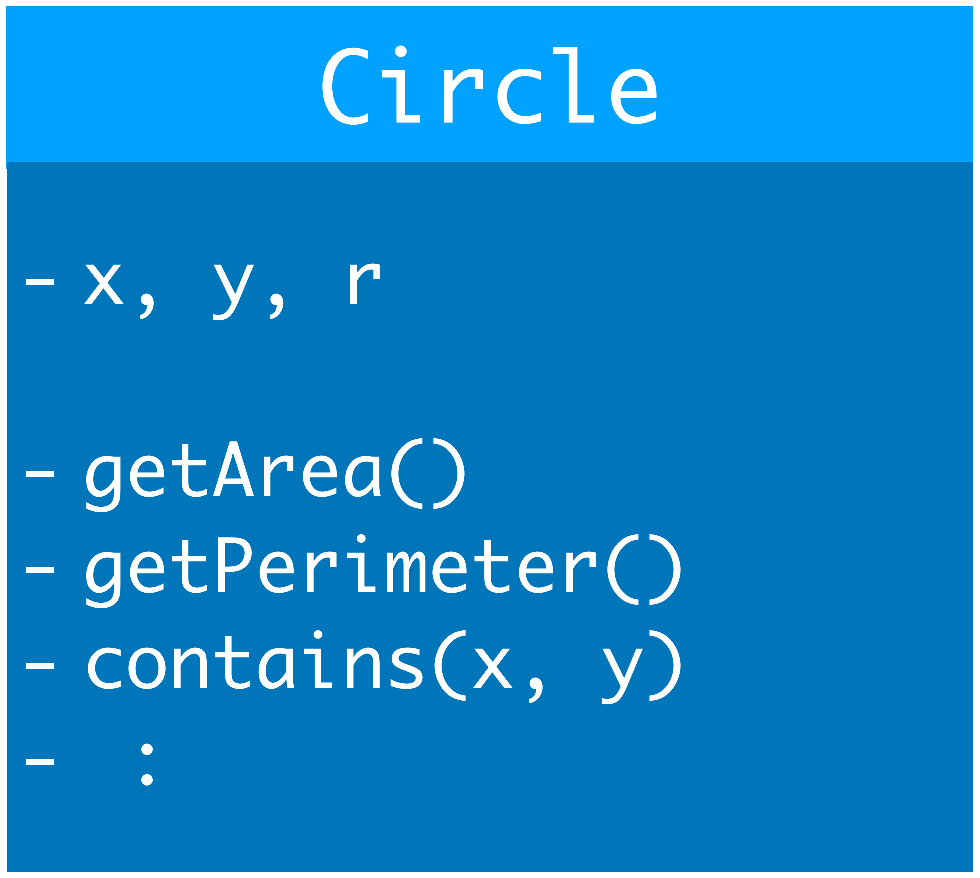
Abstraction Barrier

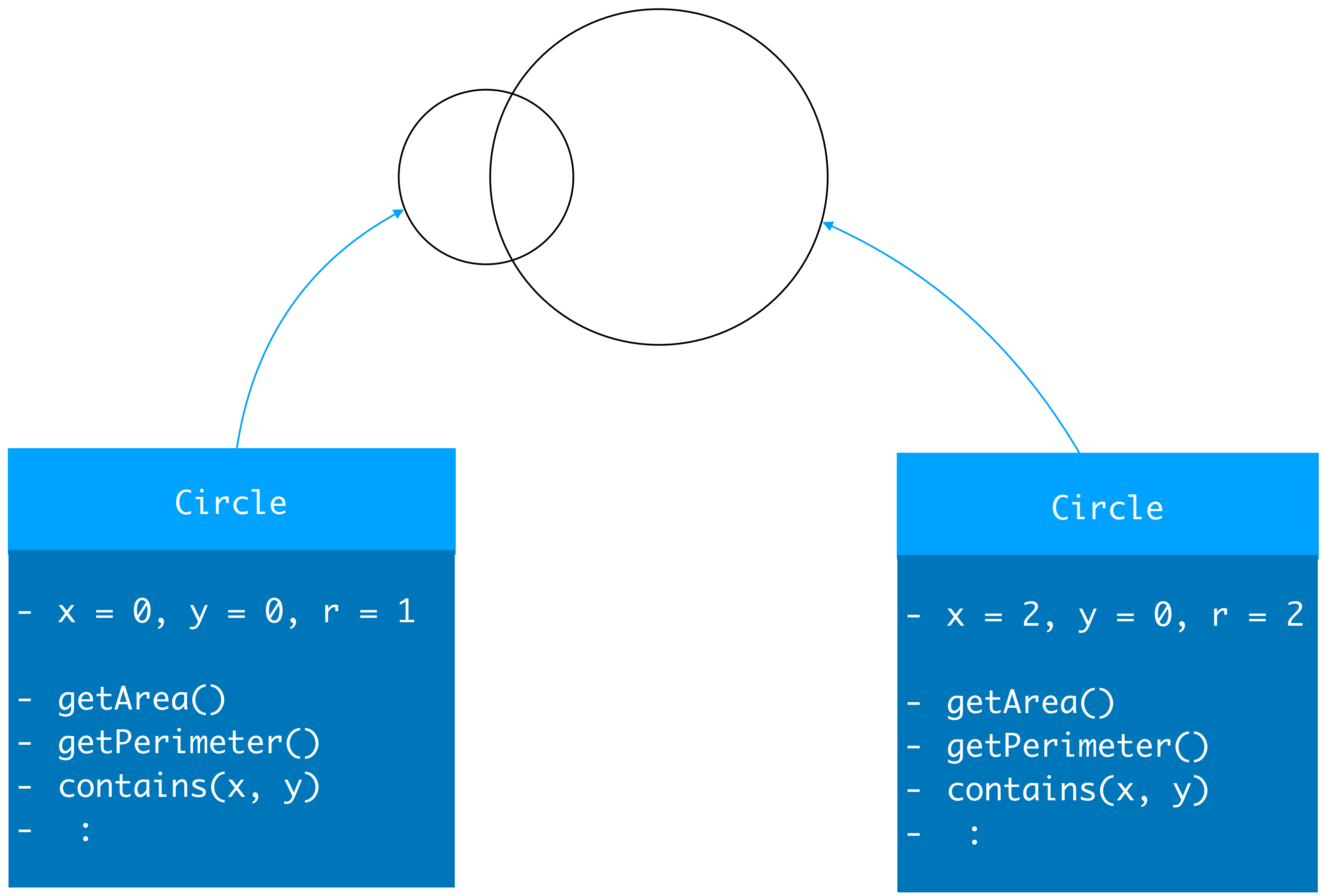


Abstraction Barrier



Class and Objects


Circle
- x, y, r
- getArea()
- getPerimeter()
- contains(x, y)
- :



Two circle objects

Data Hiding

Let's Java

Java checks for type compatibility strictly

```c
#include <stdint.h>
#include <stdio.h>

int main()
{
    printf(“%d\n”, “cs2030”);
}
```

```
#include <stdint.h>
#include <stdio.h>

int main()
{
    printf("%d\n", (int)"cs2030");
}
```

Let’s try on Java

Java ensure type safety by preventing “potentially buggy” operations on variables based on their types