Code Tracing: Sequence & Assignment

- class and object
- method
- control structure
- statement ●

06 Tracing Code by Hand
Appendix: Code Tracing Problems

cc BY NC SA

Dr James Montgomery, james.montgomery@utas.edu.au

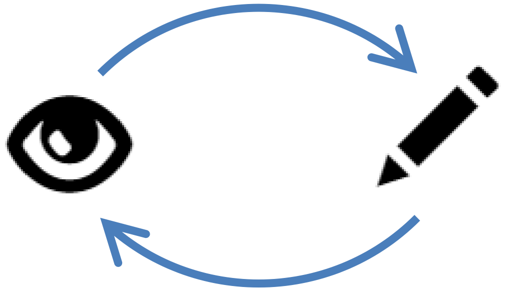
What is code tracing?

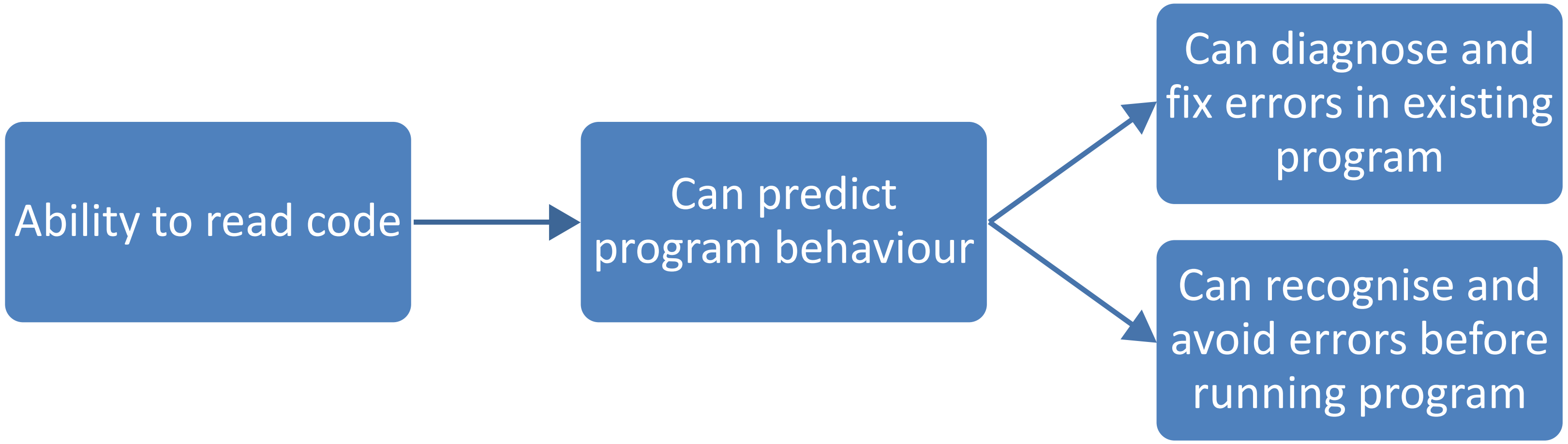
‘Tracing’, ‘desk checking’, ‘hand execution’ are synonyms for a human following the instructions in a computer and interpreting their effect

- on the **values of variables**; and
- on the **output from a program**

Why code reading is a useful skill

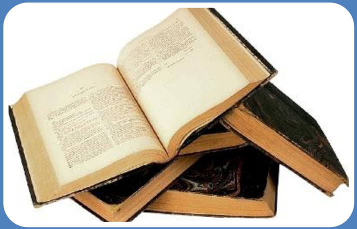
Reading and writing are complementary skills




Ability to read code
Can predict program behaviour
Can diagnose and fix errors in existing program
Can recognise and avoid errors before running program

Tracing: essential tools

Paper & pen (or text editor)



Knowledge of the programming language's semantics

- ✓ A mental model of computer memory (we will use a table holding values)
- ✓ The effect of assignment statements
- ✓ The order in which parts of a statement are executed by the computer
- ✓ The behaviour of methods that are called

If unsure about the effect of a statement: ask for help, consult documentation or write a small program to find out

Essential language semantics

```java
int a = 3;
```

① Everything in the expression on the right hand side is evaluated first

⑤ Then that value is stored in the variable on the left

```java
a = Math.max(a + 4, 6) + 20;
```

② The arguments to the method call are evaluated (7 and 6)

③ Then the method's code is executed, and eventually returns a value (7)

④ Then the addition can proceed, left to right (7 + 20 = 27)

Basic structure of a tracing table

Variables and program output

Time

Line	age	name	gender	Output
1	21			
2		Jane		
3			‘F’	
4				Jane is 21

```
     public class Example {
        public static void main(String[] args) {
1             int age = 21;
2             String name = "Jane";
3             char gender = 'F';
4             System.out.println("Jane is " + age);
          }
     }
```

High-level instructions

Read the code in sequence and ‘process’ each statement in turn

If it is a…

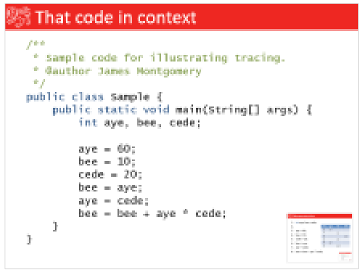
- **declaration:** add a column heading for that variable
- **assignment or initialisation:** add a row for that line and write in the variable’s new value
- **statement that produces output:** add a row for that line and record the output produced

So: Don’t have rows for lines that do neither assignment nor output, or that are not executed

Demonstration

```
1.  int aye, bee, cede;
2.
3.  aye = 60;
4.  bee = 10;
5.  cede = 20;
6.  bee = aye;
7.  aye = cede;
8.  bee = bee + aye * cede;
```

Line	aye	bee	cede
3	60		
4		10	
5			20
6		60	
7	20		
8		460	



That code in context

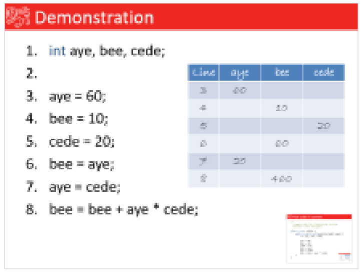
```
/**
 * Sample code for illustrating tracing.
 * @author James Montgomery
 */
public class Sample {
    public static void main(String[] args) {
        int aye, bee, cede;

        aye = 60;
        bee = 10;
        cede = 20;
        bee = aye;
        aye = cede;
        bee = bee + aye * cede;
    }
}
```

That code in context

```
/**
 * Sample code for illustrating tracing.
 * @author James Montgomery
 */
public class Sample {
    public static void main(String[] args) {
        int aye, bee, cede;

        aye = 60;
        bee = 10;
        cede = 20;
        bee = aye;
        aye = cede;
        bee = bee + aye * cede;
    }
}
```



Demonstration

```
1. int aye, bee, cede;
2.
3. aye = 60;
4. bee = 10;
5. cede = 20;
6. bee = aye;
7. aye = cede;
8. bee = bee + aye * cede;
```

Line	aye	bee	cede
3	60		
4		10	
5			20
6		60	
7	20		
8		460	

Code Tracing: Selection & Repetition

This material is relevant *after* you have learned about conditional statements and loops

class and object
method
control structure
statement ●

06 Tracing Code by Hand
08 Making Decisions
09 Repeating Actions with Loops
Appendix: Code Tracing Problems

BY NC SA

Dr James Montgomery, james.montgomery@utas.edu.au

High-level instructions (again)

Read the code in sequence and 'process' each statement in turn

If it is a…

- **declaration:** add a column heading for that variable
- **assignment or initialisation:** add a row for that line and write in the variable's new value
- **statement that produces output:** add a row for that line and record the output produced

So: Don't have rows for lines that do neither assignment nor output, or that are not executed

And if a line is repeated then add another row for it

Demonstration

```
1.  int a = 2;
2.
3.  if (a == 3) {
4.      a = 0;
5.  } else {
6.      a = 10;
7.  }
```

Line	a
1	2
6	10

← You might choose to write the outcome of the condition at line 3, if it helps you follow the code, but it's not required

Demonstration

```
1.  int a = 0;
2.
3.  while (a <= 3) {
4.      a++;
5.  }
6.  System.out.println("a: " + a);
```

Line	a	Output
1	0	
4	1	
4	2	
4	3	
4	4	
6		a: 4

If it helps you then include the outcome of the loop condition, but you'll generally find it's not necessary (as you can check it using the values in the table)