# ENGG1003 - Friday Week 1

## Algorithms and Pseudocode

Brenton Schulz

University of Newcastle

February 17, 2019

# Algorithms

- Informally, an *algorithm* is a series of steps which accomplishes a task
- More accurately, the steps (instructions) must:
  - Have a strict order
  - Be unambiguous
  - Be executable
- "Executable" means that the *target platform* is capable of performing that task.
  - eg: An industrial welding robot can execute "move welding tip 1 cm left". A mobile phone can't.

# Algorithms

- An algorithm exists purely as an abstract concept until it is communicated
- We will use:
  - *Pseudocode* to communicate algorithms to ourselve's and other people
  - The languages C and MATLAB to communicate algorithms to computers
- Pseudocode can be very formal, as engineers we will only use formal rules if required
  - eg: When documenting algorithms for other people
  - Your own "working out" can be anything that helps *you*

# Algorithm Example 1

**Example 1:** Algorithm given to mum to start my car (2015 Tarago)
**Result:**The vehicle's engine is idling
**Initialisation:** stand next to the vehicle, key fob in hand

1. Depress the unlock button on the key fob, car will beep twice
2. Place key fob in your pocket
3. Enter the vehicle, sit in the driver's seat
4. Ensure that the gear selector has P engaged
5. Depress the brake pedal
6. Observe that the green LED is lit on the engine start button
7. Press the engine start button
8. If engine is not idling
   - Call me

# Example Discussion

- Algorithms typically need to feel over-explained
  - Computers are *really stupid*; get in the habit of over-thinking everything
- The algorithm contained *flow control*
  - The final step ("call me") was *conditional* on the car not starting
- To study conditions we will discuss *Boolean algebra* later, but first...

# Algorithm Example 2

A wife asks her husband, a programmer, "Could you please go shopping for me and buy one carton of milk, and if they have eggs, get 6?"

A short time later the husband comes back with 6 cartons of milk and his wife asks, "Why did you buy 6 cartons of milk?"

He replies, "They had eggs."

# Algorithm Example 2a

*Lets make this more realistic.*

A wife asks her robot helper, "Could you please go shopping for me and buy one carton of milk, and if they have eggs, get 6?"

The robot replies: "Unknown instruction: 'get 6'. "

# Boolean Algebra Basics

- Computers don't understand "maybe"
- A *condition* must be absolutely **true** or **false**
- Boolean algebra (or Boolean logic) is a field of mathematics which evaluates *logical statements* as either true or false
- Boolean *variables* can only take the values **true** (or 1) or **false** (or 0)
- Boolean algebra defines three *operators*:
  - OR
  - AND
  - NOT

# Boolean Algebra Basics

- Boolean variables can be allocated any symbols (just like in "normal" algebra)
  - Typically get uppercase letters
  - eg: X = A OR B
- Various symbols can be used for OR/AND/NOT, we will only use the words here
  - Write them in capitals to remove ambiguity
  - C and MATLAB have their own symbols for Boolean algebra
  - Other courses (eg: ELE17100) will use others again

# Boolean Operators

- An *operand* is a value on which a mathematical operation takes place
  - eg: In "1 + 2" the 1 and 2 are operands and + is the operator
- OR - Evaluates true if either operand is true
  - X = A OR B
  - X is true if A or B is true
- AND- Evaluates true only when *both* operands are true
  - X = A AND B
  - X is true only if both A and B is true

# Boolean Operators

- Observe that OR and AND are *binary* operators
  - They operate on two operands
  - From latin "bini" meaning "two together"
- The NOT operator is *unitary*
  - ie: it only operates on *one* operand
  - NB: The operand could be a single variable or complex expression
- NOT performs a logical inversion
  - NOT true = false
  - NOT false = true

# Algorithm Example 3 - Quadratic Root Finding

From high school you should know that the equation

$$ax^2 + bx + c = 0 \tag{1}$$

has solutions given by

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{2}$$

lets write an algorithm which only deals with real numbers.

# Algorithm Example 3 - Quadratic Root Finding

**Input:** Real numbers $a$, $b$, and $c$
**Output:** Three numbers:

1. The number of solutions, N
2. One of the roots, $x_1$
3. The other root, $x_2$

**Behaviour:**

- If N is 2 then $x_1$ and $x_2$ are different real numbers
- If N is 1 then $x_1$ is the unique solution and $x_2$ is undefined
- If N is 0 then $x_1$ and $x_2$ are undefined

# Algorithm Example 3 - Quadratic Root Finding

BEGIN
  $D = \sqrt{b^2 - 4ac}$
  IF $D < 0$
    $N = 0$
  ELSE IF $D = 0$
    $N = 1$
    $x_1 = \frac{-b}{2a}$
  ELSE IF $D > 0$
    $N = 2$
    $x_1 = \frac{-b+D}{2a}$
    $x_2 = \frac{-b-D}{2a}$
  ENDIF
END

- The IF ... ELSE IF flow control construct forces exclusive execution of only *one* block
- The first condition that is true causes execution of that block
- Subsequent blocks ignored

# C listing template

```
1 #include <stdio.h>
2 int main() {
3   printf("Custom lstlisting template\n");
4 }
```

```
#include <stdio.h>
int main() {
        printf("default C style\n");
}
```

# Columns Template

left side

right side