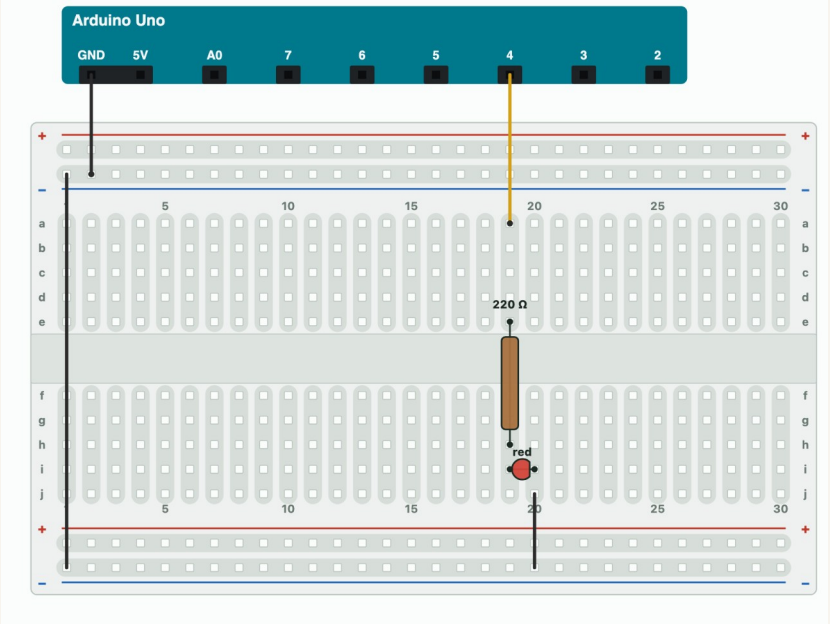
# Daily Review



1. **State** how long **delay(1000)** waits. (**value and unit**)
2. **State** how often **setup()** runs.
3. **State** how often **loop()** runs.
4. **State** what **digitalWrite(4, HIGH);** does.
5. **State** how long **pause 500** waits. (**value and unit**)

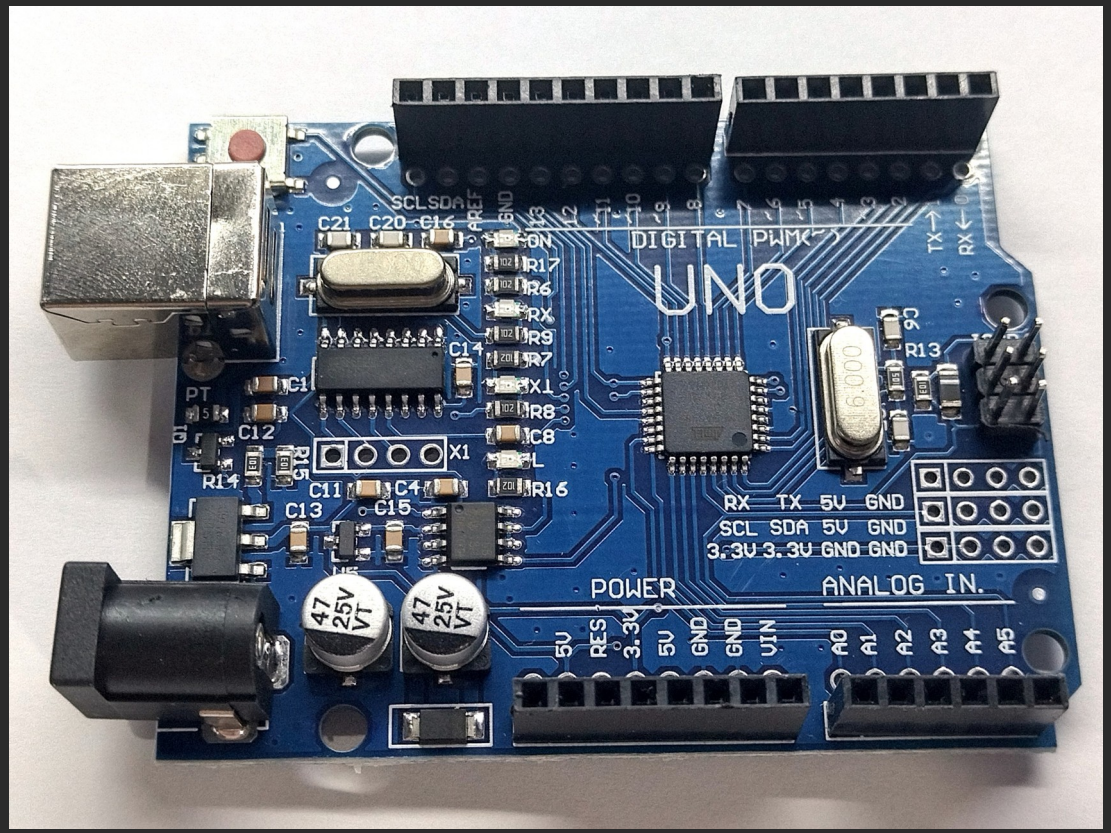
# Build, Test and Fix

Put the program on a real Uno. Then make it work.




National 5 Engineering Science

# This board has no program on it yet. What must you give it to flash an LED?




An Arduino Uno

PART 1

# Build It

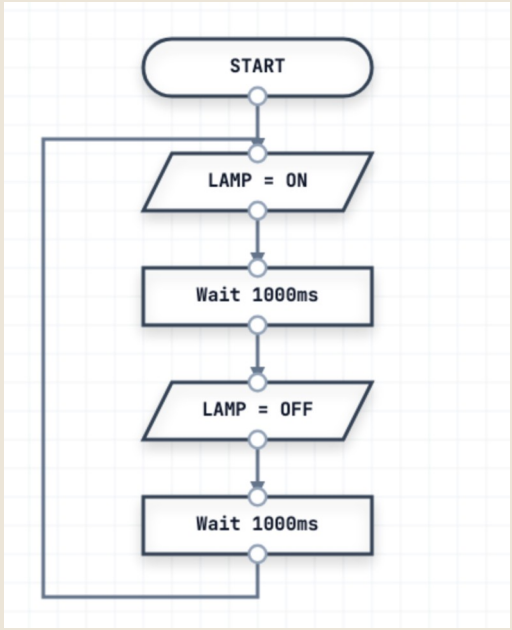
Simulate, verify, upload, build.

# By the end of this part…

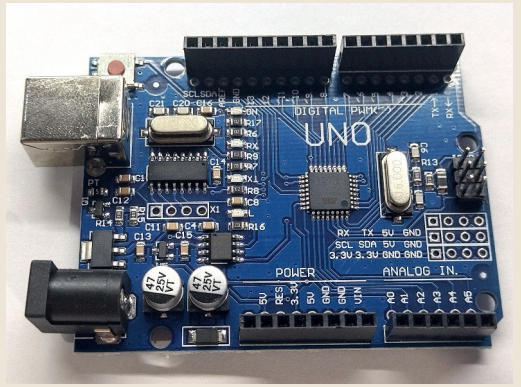


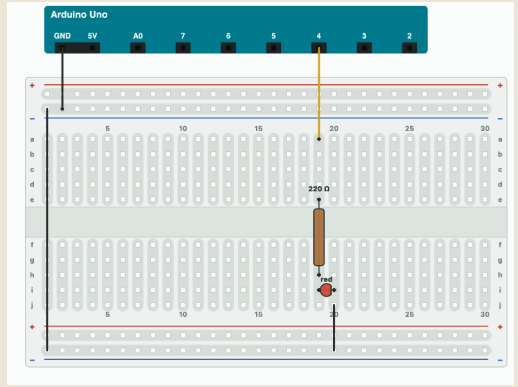
1. I can **wire** an **LED** to a **pin** on the Uno
2. I can **match** each **line of code** to a **flowchart box**
3. I can **upload** a program and **check** it works

# From Sim to Board




Generate
Code
Duino.app
Verify




1 Sim

2 Verify

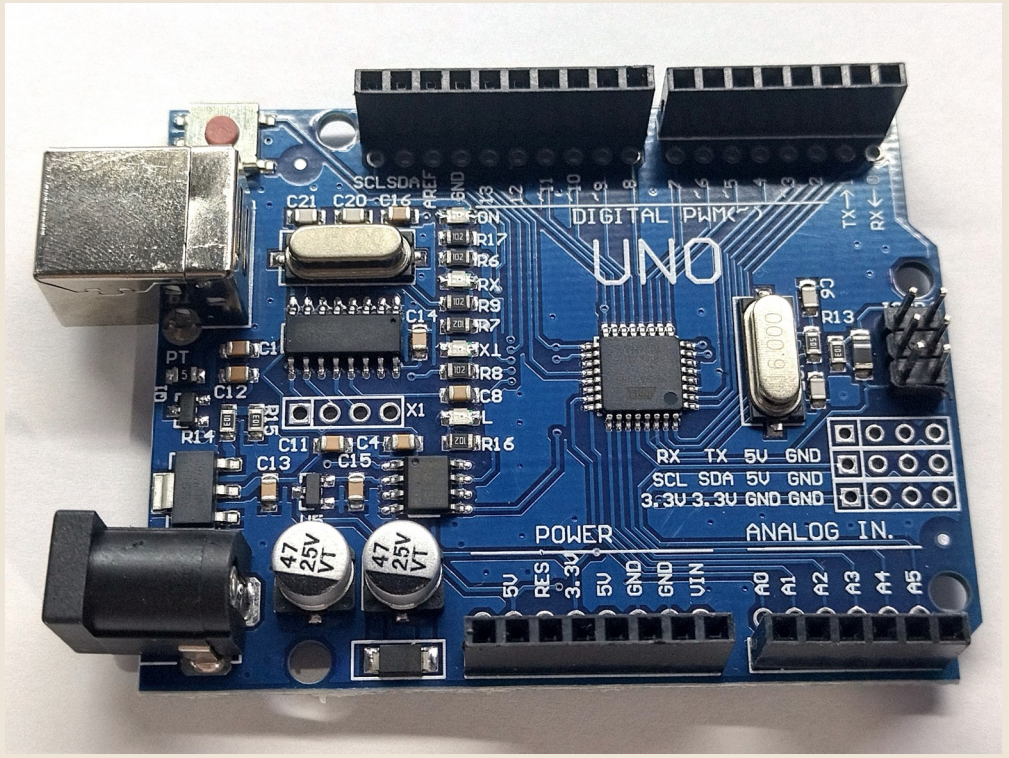
3 Upload

4 Build

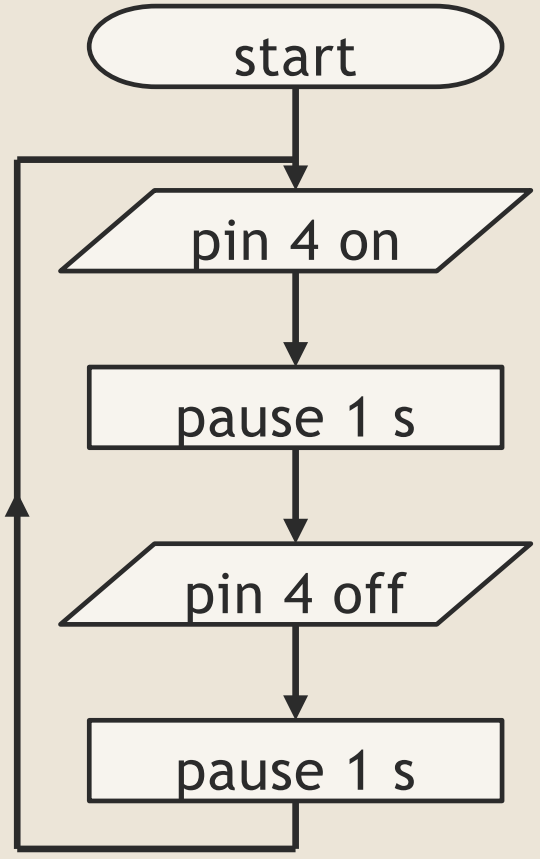
# What to Collect

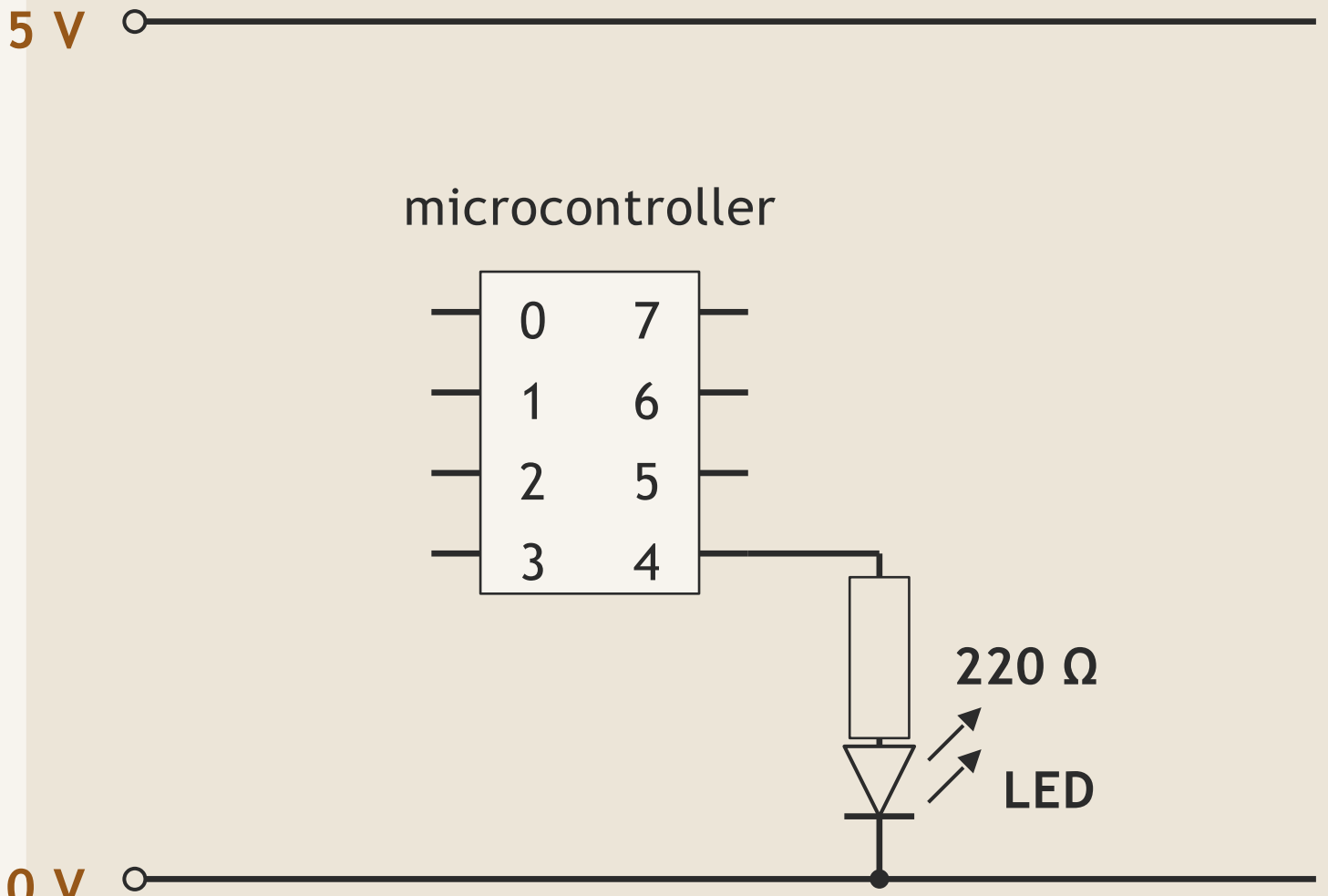
Collect one set per pupil.

1 Uno and its USB lead
2 Breadboard
3 3 LEDs: red, amber, green
4 3 × 220 Ω resistors
5 6 jumper leads



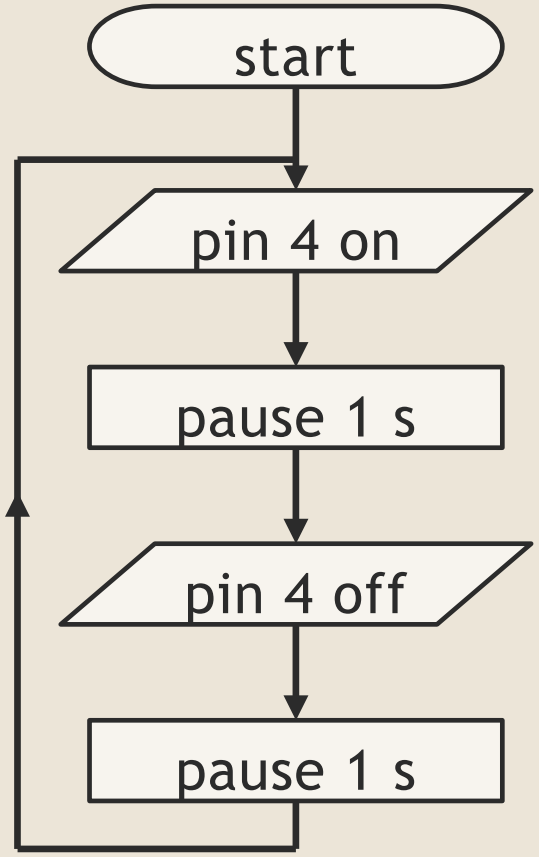
# Worked Example 1: The Plan and the Wires



# Worked Example 1: The Code

```
1   void setup() {
2     pinMode(4, OUTPUT);
3   }
4   void loop() {
5     digitalWrite(4, HIGH);
6     delay(1000);
7     digitalWrite(4, LOW);
8     delay(1000);
9   }
```



# Worked Example 1, Both Ways

## PSEUDOCODE

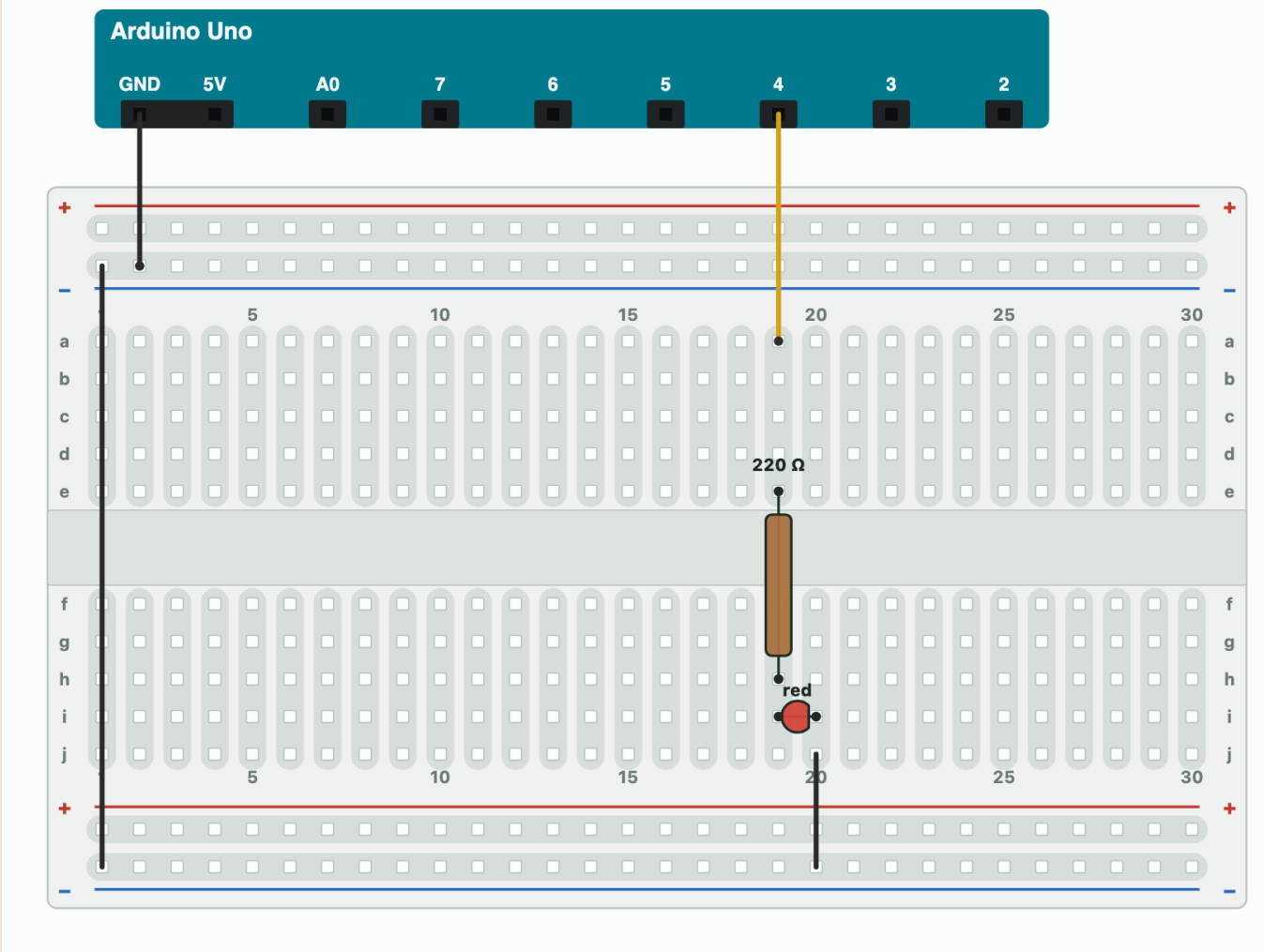
Flowchart

```
1 main:   switch on 4
2         pause 1000
3         switch off 4
4         pause 1000
5         goto main
```

## ARDUINO C++

```
1 void setup() {
2   pinMode(4, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(4, HIGH);
6   delay(1000);
7   digitalWrite(4, LOW);
8   delay(1000);
9 }
```

# Worked Example 1: Build It




**Yellow lead**

pin 4 to the 220 Ω

**LED**

long leg to the resistor

**Black lead**

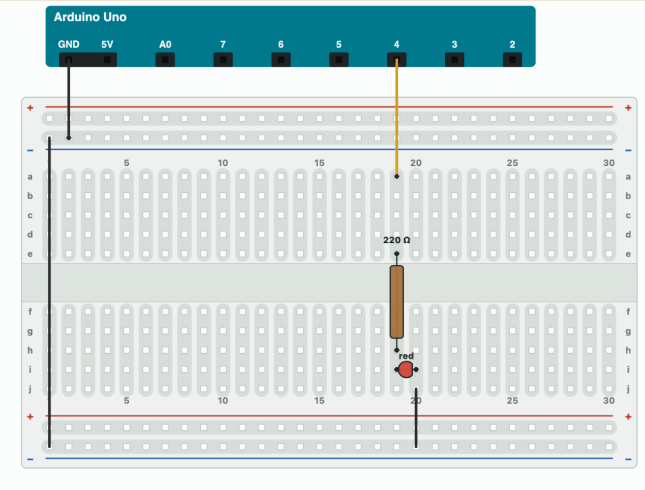
GND to the – rail

# Build It: Worked Example 1

## Practical sheet 1

**Build** it: NoSyntaxSim → Duino.app → **Verify**.

**Upload**, **unplug**, **wire** it. **Check**: 1 s on, 1 s off.



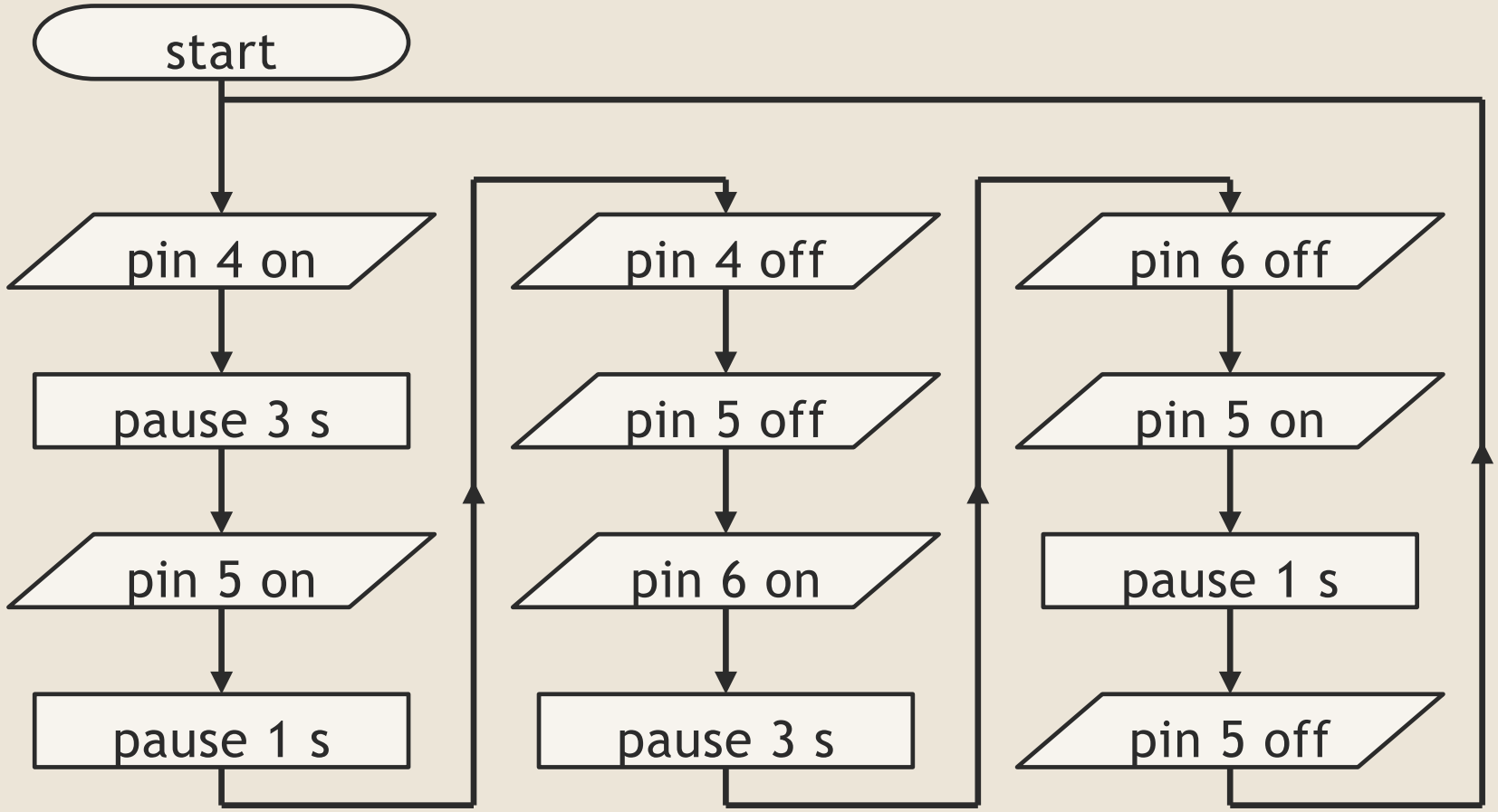

**SAFETY** USB power only. Unplug before you move a wire.

# Try Changing This: Worked Example 1



1. Both delays become **delay(200)**. **State** whether it flashes **faster** or **slower**.
2. The delays become **delay(100)** then **delay(900)**. **Describe** what the **LED** does.
3. The LED moves to **pin 5**. **State** what else must **change**.

# Worked Example 2: Traffic Lights



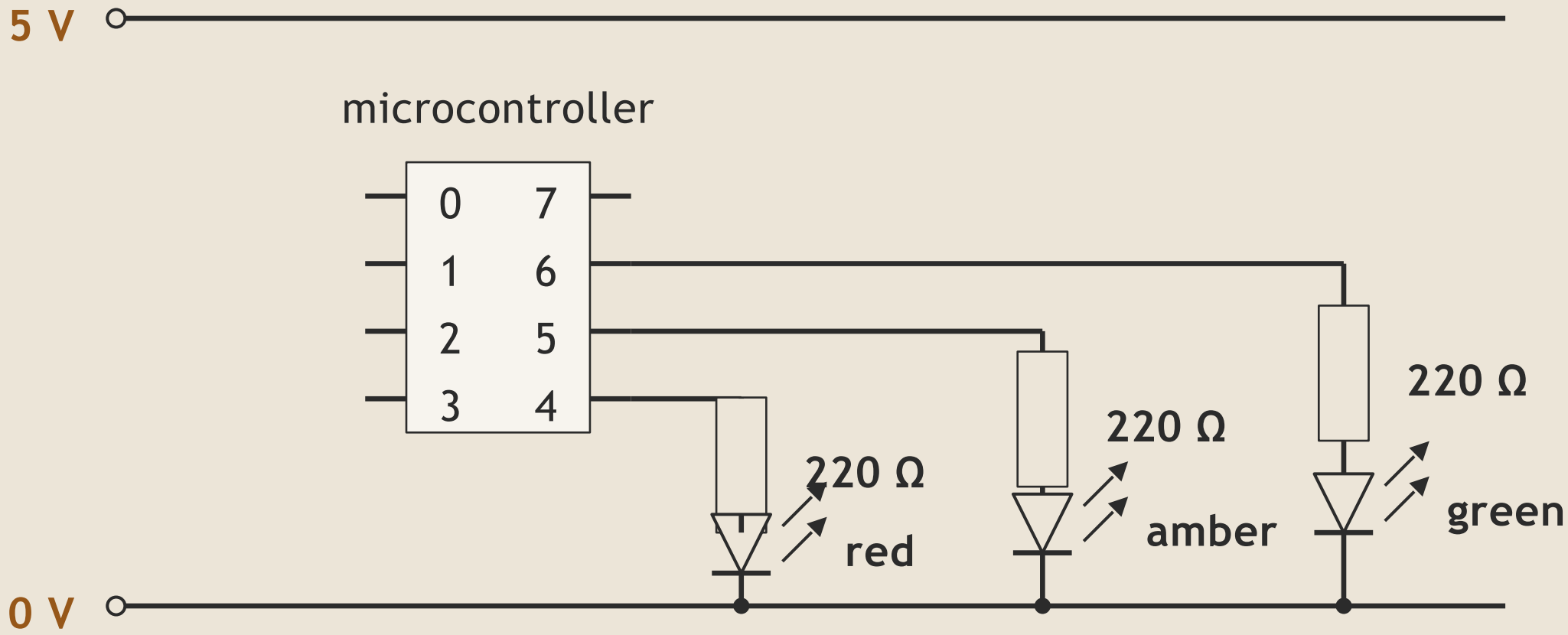

1 red

2 red + amber

3 green

4 amber

# Worked Example 2: The Wires




```
pinMode(4, OUTPUT);  pinMode(5, OUTPUT);  pinMode(6, OUTPUT);
```

# Worked Example 2: The Code

```
void loop() {
  digitalWrite(4, HIGH);
  delay(3000);
  digitalWrite(5, HIGH);
  delay(1000);
  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  digitalWrite(6, HIGH);
  delay(3000);
  digitalWrite(6, LOW);
  digitalWrite(5, HIGH);
  delay(1000);
  digitalWrite(5, LOW);
}
```

# Worked Example 2, Both Ways

## PSEUDOCODE

Flowchart

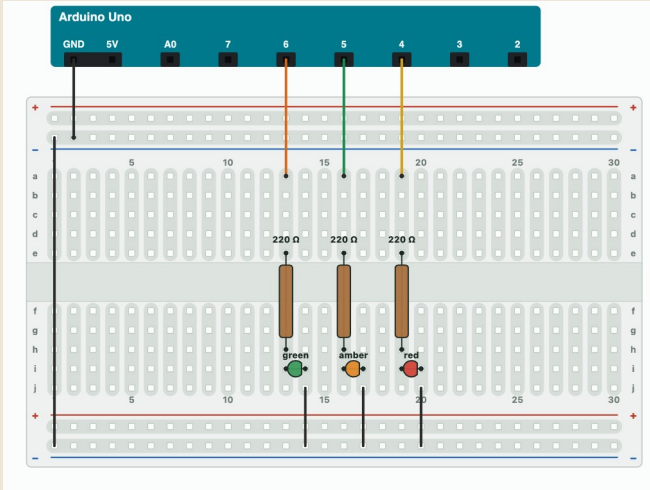
```
1   main:   switch on 4
2           pause 3000
3           switch on 5
4           pause 1000
5           switch off 4
6           switch off 5
7           switch on 6
8           pause 3000
9           switch off 6
10          switch on 5
11          pause 1000
12          switch off 5
13          goto main
```

## ARDUINO C++

```
1   void setup() {
2     pinMode(4, OUTPUT);
3     pinMode(5, OUTPUT);
4     pinMode(6, OUTPUT);
5   }
6   void loop() {
7     digitalWrite(4, HIGH);
8     delay(3000);
9     digitalWrite(5, HIGH);
10    delay(1000);
11    digitalWrite(4, LOW);
12    digitalWrite(5, LOW);
13    digitalWrite(6, HIGH);
14    delay(3000);
15    digitalWrite(6, LOW);
16    digitalWrite(5, HIGH);
17    delay(1000);
18    digitalWrite(5, LOW);
19  }
```

# Build It: Worked Example 2

## Practical sheet 1




**Build** it: NoSyntaxSim → Duino.app → **Verify**.

**Upload**, **unplug**, **add** amber and green.

**SAFETY** USB power only. Unplug before you move a wire.

# Try Changing This: Worked Example 2




1. Green must stay on for **6 s**, not 3 s. **State** the **line** to change, and how.
2. You add **// by** and your name. **State** whether it **still runs**, and why.
3. The 4 and 6 swap places **in setup() only**. **Describe** what **changes**.

# Checkpoint




1. **Explain** why each LED has a **220 Ω resistor**.
2. A pin is **not set** in **setup()**. **Describe** what the **LED** does.
3. Two LEDs come **on together**. **State** how the **flowchart** shows this.

# Plenary: Part 1




1. **State** what **setup()** sets each pin as.
2. **State** what the closing } of **loop()** does in the **flowchart**.
3. **State** how long **delay(3000)** waits. (**value and unit**)

# Plenary: Part 1




4 **State** what the **pin** in the **code** must match.

5 The code passes **Verify**. **State** the **next step**.

# Look back — by the end of this part…



1. I can **wire** an **LED** to a **pin** on the Uno
2. I can **match** each **line of code** to a **flowchart box**
3. I can **upload** a program and **check** it works

# Daily Review



1. **State** where **pinMode(5, OUTPUT);** goes.
2. **State** how long **delay(250)** waits. (**value and unit**)
3. **State** how many **output boxes** light **two LEDs**.
4. **State** what the **220 Ω** resistor **protects**.
5. The wire is in **pin 4**; the code says **pin 5**. **State** what the LED does.

PART 2

# Sensors and Faults

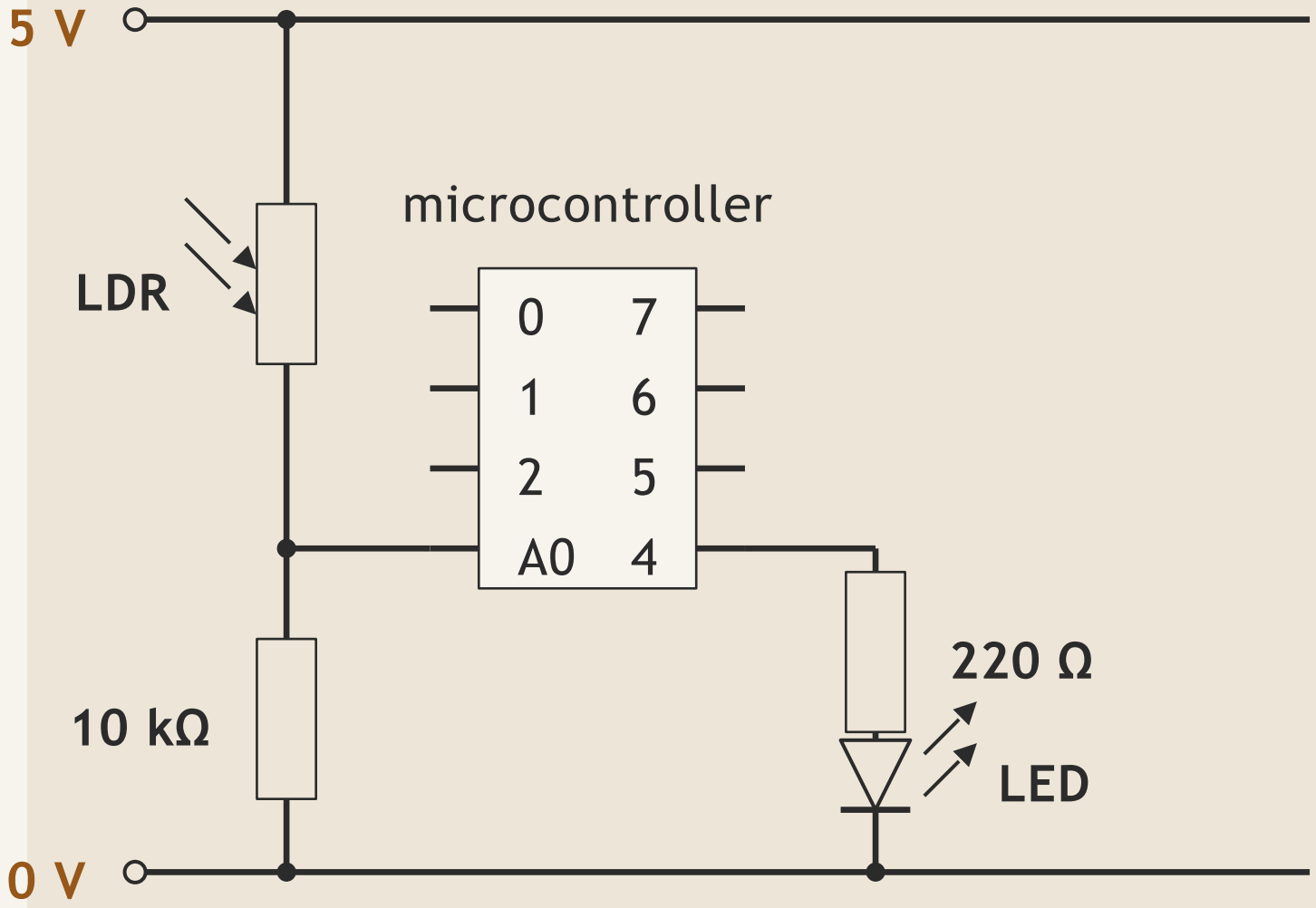
Make it see the dark. Then find what is wrong.

# By the end of this part…



1. I can **read** a **sensor** with **analogRead()**
2. I can **count** a **fixed loop** in Arduino C++
3. I can **find** and **fix faults** in a program

# Worked Example 3: The LDR on A0




**analogRead(A0)**

a number from 0 to 1023

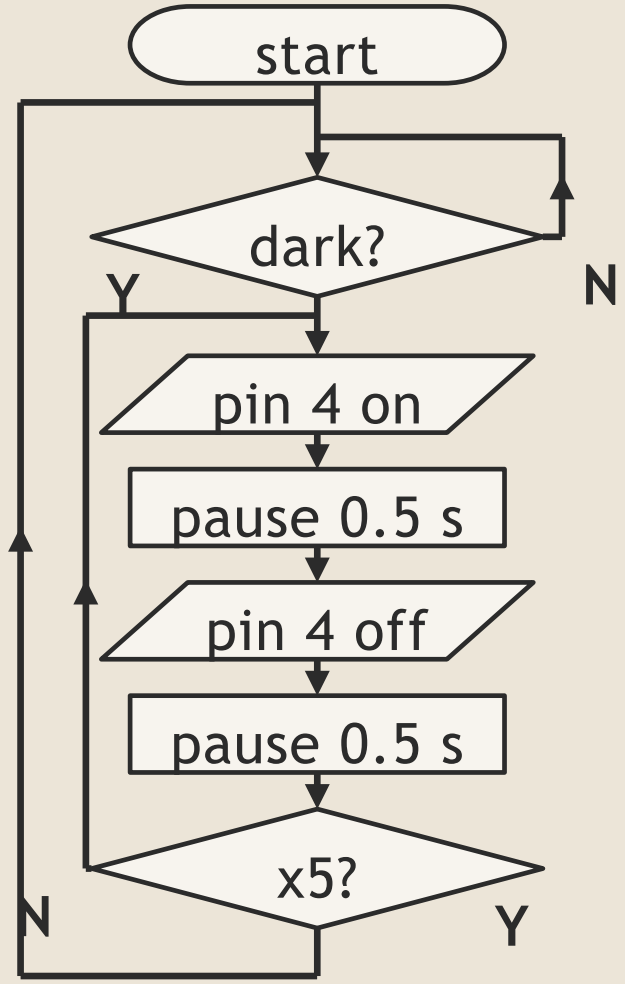
**Dark**

LDR resistance up, A0 down

**light < 300**

means it is dark

# Worked Example 3: Flowchart and Code




## ARDUINO C++

```
1  void setup() {
2    pinMode(4, OUTPUT);
3  }
4  void loop() {
5    while (analogRead(A0) >= 300) { }
6    for (int count = 0; count < 5; count++) {
7      digitalWrite(4, HIGH);
8      delay(500);
9      digitalWrite(4, LOW);
10     delay(500);
11   }
12 }
```

# Worked Example 3, Both Ways

## PSEUDOCODE

Flowchart

```
1   main:     let light = A0
2             if light >= 300 then main
3             let count = 0
4   label_1:  switch on 4
5             pause 500
6             switch off 4
7             pause 500
8             let count = count + 1
9             if count = 5 then main
10            goto label_1
```

## ARDUINO C++

```
1   void setup() {
2     pinMode(4, OUTPUT);
3   }
4   void loop() {
5     while (analogRead(A0) >= 300) { }
6     for (int count = 0; count < 5; count++) {
7       digitalWrite(4, HIGH);
8       delay(500);
9       digitalWrite(4, LOW);
10      delay(500);
11    }
12  }
```

# Counting From 0




**for (int count = 0;**

count < 5; count++)

**count = 0**

start at 0

**count < 5**

go on while under 5

**count++**

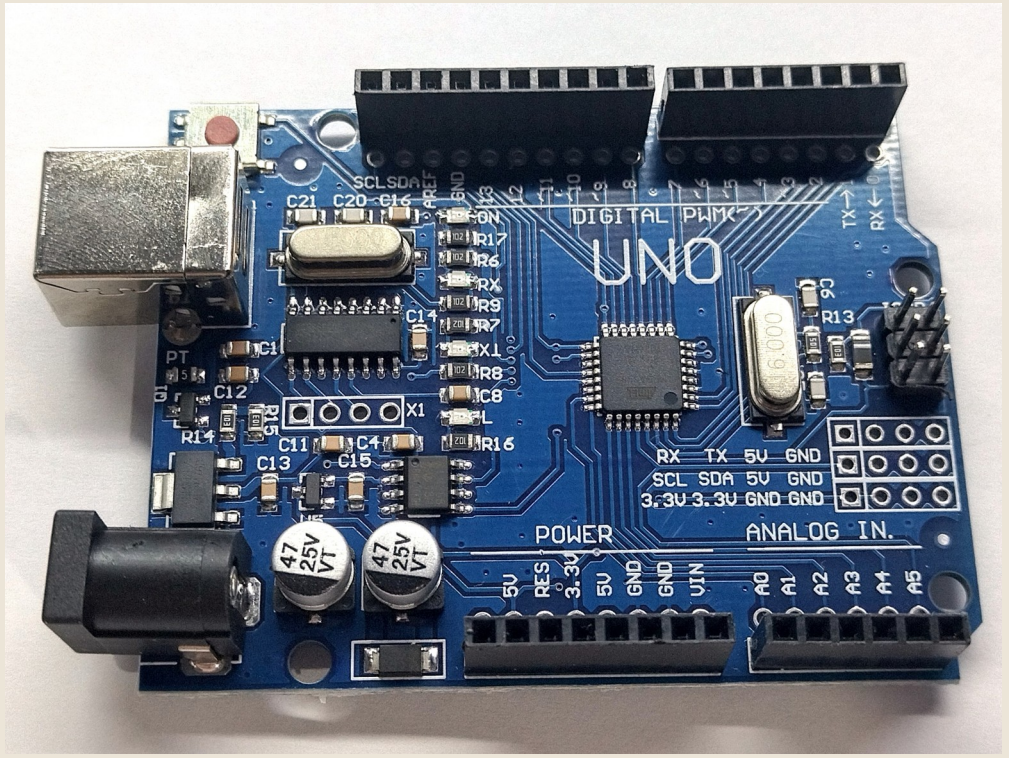
add 1 each pass

| count | count < 5? | flash |
|---|---|---|
| 0 | yes | 1 |
| 1 | yes | 2 |
| 2 | yes | 3 |
| 3 | yes | 4 |
| 4 | yes | 5 |
| 5 | no | stop |

# What to Collect

Collect one set per pupil.

1 LDR
2 2 × 10 kΩ (brown, black, orange)
3 Push button
4 Buzzer
5 4 more jumper leads

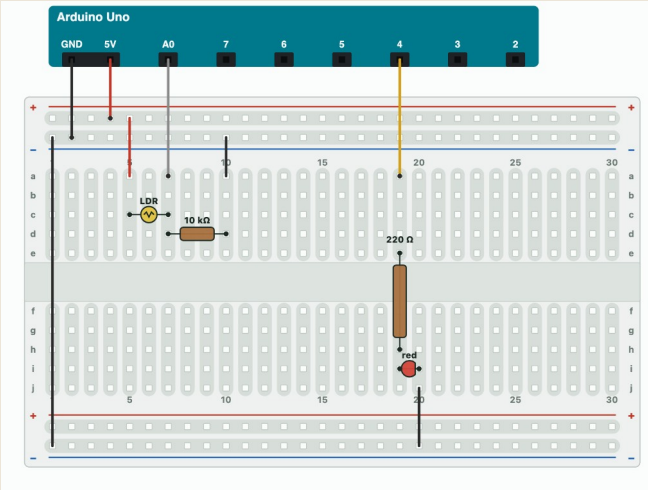


# Build It: Worked Example 3

## Practical sheet 2

**Build** it: NoSyntaxSim → Duino.app → **Verify**.

**Upload**, **unplug**, **add** the **LDR** (page 18).




**SAFETY** USB power only. Unplug before you move a wire.

# Try Changing This: Worked Example 3



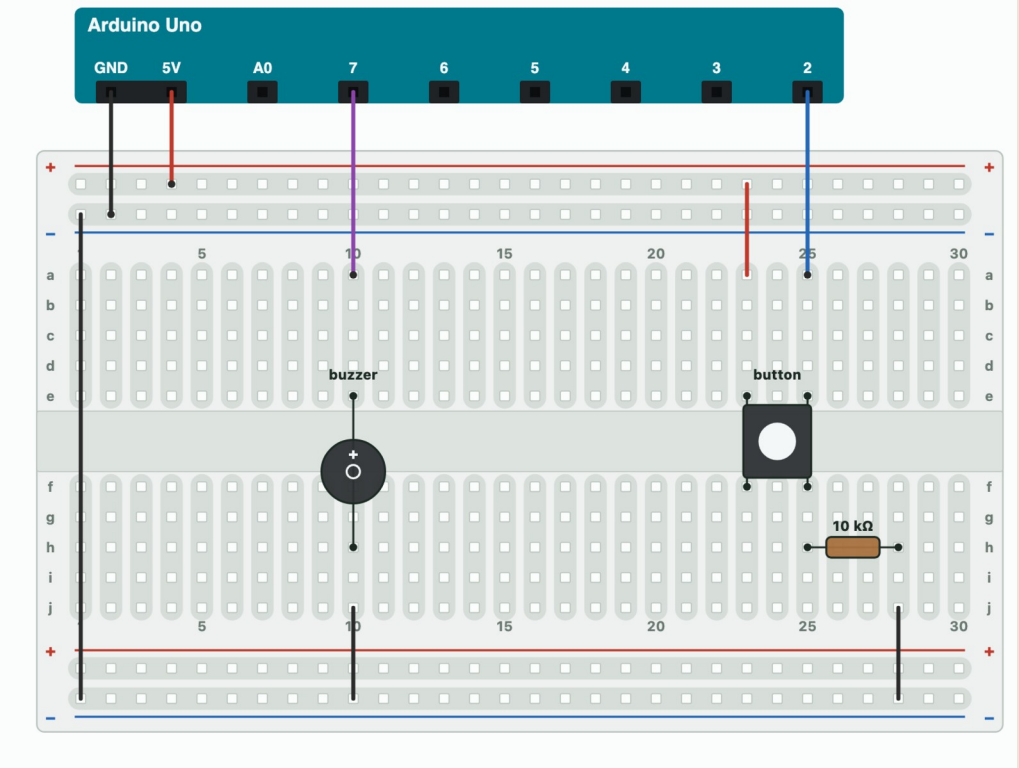

1. **300** becomes **600**. **State** whether it is **easier** or **harder** to set off.

2. **count < 5** becomes **count < 10**. **State** how many **flashes** it gives.

3. **>=** becomes **<**. **State** when it **flashes**.

# Debugging: The Method

1. What should it do?
2. What does it do?
3. Which line?
4. Change one thing
5. Test it again



# The Faults to Look For




1. Verify says **expected ';'. State** the **fix.**
2. A **{** has **no }. State** the **fix.**
3. **Correct** this line: **digitalwrite(4, high);**

# The Faults to Look For




4. **Correct** this line: **if (digitalRead(2) = HIGH)**

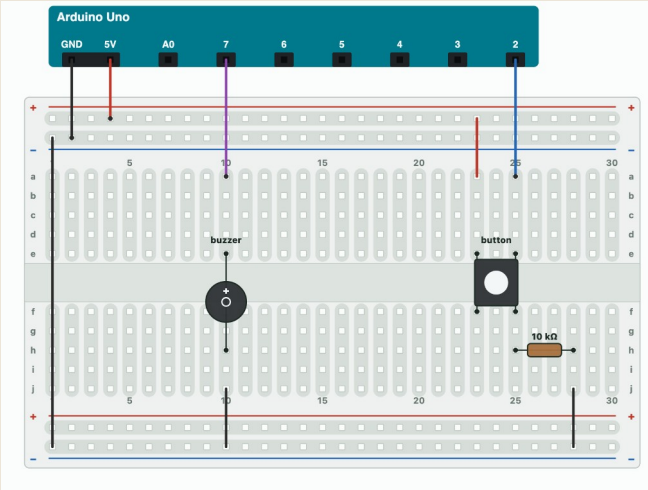
5. **delay(1)** is meant to wait **1 s**.
**Correct** it.

6. The LED **never lights**. **State** the **line** to check.

# Find and Fix: Debug Builds 1 and 2

## Practical sheet 2

**Type** each into Duino.app. **Verify**, **Upload**.
**Wire** it. **Find** the **2 faults** in each.
**Re-test** after each fix.




**SAFETY** USB power only. Unplug before you move a wire.

# Find and Fix: Debug Build 1



```
1   void setup() {
2   }
3   void loop() {
4     digitalWrite(4, HIGH);
5     delay(1);
6     digitalWrite(4, LOW);
7     delay(1000);
8   }
```

It **should flash pin 4**: 1 s on, 1 s off. **Find** the **two faults.**

1 **Line 2**

2 **Line 5**

# Debug Build 1: Fixed, Both Ways

## PSEUDOCODE

Flowchart

```
1 main:   switch on 4
2         pause 1000
3         switch off 4
4         pause 1000
5         goto main
```

## ARDUINO C++

```
1 void setup() {
2   pinMode(4, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(4, HIGH);
6   delay(1000);
7   digitalWrite(4, LOW);
8   delay(1000);
9 }
```

# Find and Fix: Debug Build 2



```
1   void setup() {
2     pinMode(2, INPUT);
3     pinMode(7, OUTPUT);
4   }
5   void loop() {
6     if (digitalRead(2) = HIGH) {
7       digitalWrite(7, HIGH);
8       delay(2000);
9       digitalWrite(7, LOW);
10    }
```

It should **buzz 2 s** when **pressed. Find** the **two faults.**

1 **Line 6**

2 **Line 11**

# Debug Build 2: Fixed, Both Ways

## PSEUDOCODE

Flowchart

```
main:   if input2 is off then main
        switch on 7
        pause 2000
        switch off 7
        goto main
```

## ARDUINO C++

```
void setup() {
  pinMode(2, INPUT);
  pinMode(7, OUTPUT);
}
void loop() {
  if (digitalRead(2) == HIGH) {
    digitalWrite(7, HIGH);
    delay(2000);
    digitalWrite(7, LOW);
  }
}
```

# Watch Out: The Exam Never Asks for Arduino C++

**THE MISTAKE**

- digitalWrite(4, HIGH);
- delay(1000);
- Arduino C++ in an exam answer.

**THE CORRECT ANSWER**

- pin 4 on
- pause 1 s
- A pin number and a unit.

# Checkpoint




1. **State** the **range** of numbers **analogRead(A0)** gives.
2. A **for** loop has **count < 3**. **State** how many times it **runs**.
3. **Explain** why you **change one thing** at a time.

# Plenary: Part 2




1. **State** what happens to **A0's reading** in the **dark**.
2. **State** which sign **compares** in an **if**: = or ==.
3. **State** what every **{** needs.

# Plenary: Part 2




4 **State** how the flowchart shows a **fixed loop**.

5 **State** what you **never write** in an **exam** answer.

# Look back — by the end of this part…



1. I can **read** a **sensor** with **analogRead()**
2. I can **count** a **fixed loop** in Arduino C++
3. I can **find** and **fix faults** in a program

# Daily Review



1. **State** how long **delay(1)** waits. (**value and unit**)
2. **State** the sign that **compares** in an **if**.
3. **State** whether **A0** needs a **pinMode** line.
4. **State** step 4 of the **debugging method**.
5. **State** what an **exam** answer **never** uses.

PART 3

# Test, Amend, Re-test

It works once. Does it do every job?

# By the end of this part…



1. I can **write a test** for **each job** a program does
2. I can **say what happened**, not just 'it failed'
3. I can **plan a change** and **test** it **again**

# Why Engineers Test

**Job 1**

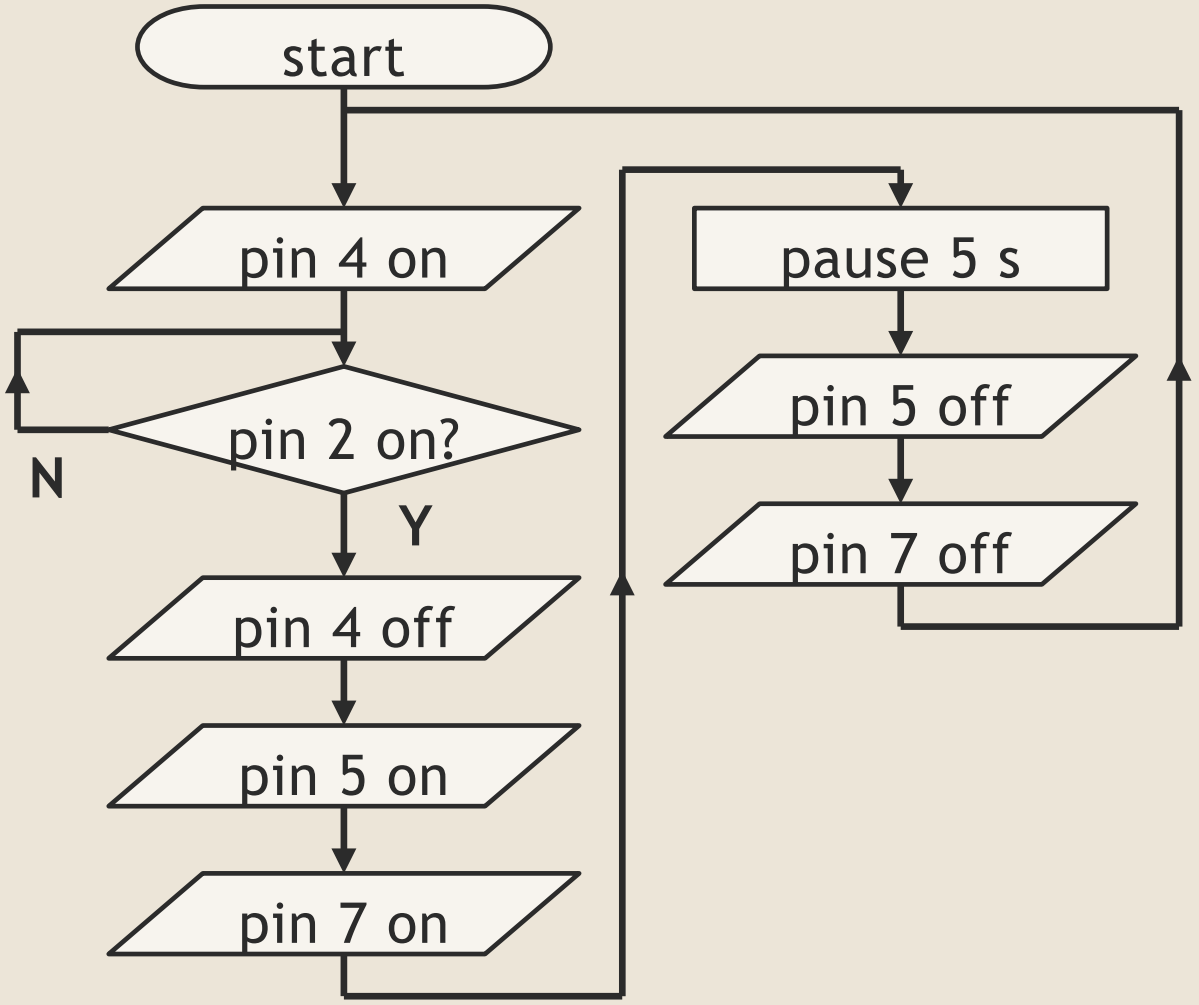
red until the button is pressed

**Job 2**

green and beeper for 5 s

**Job 3**

back to red, beeper off



# The Test Table



| Test | Should happen | Initial result | Planned amendment | Re-test |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |

Initial result: what did happen · Amendment: the change

# Worked Example: The Crossing

| Test | Should happen | Initial result | Planned amendment | Re-test |
|---|---|---|---|---|
| Press | Red off, green on | Red off, green on | None | |
| Wait 5 s | Green off, red on | Green off, red on | None | |
| Beeper | Stops after 5 s | Stays on | Add pin 7 off | Stops: pass |

# The Fault, in the Sim

## Last box

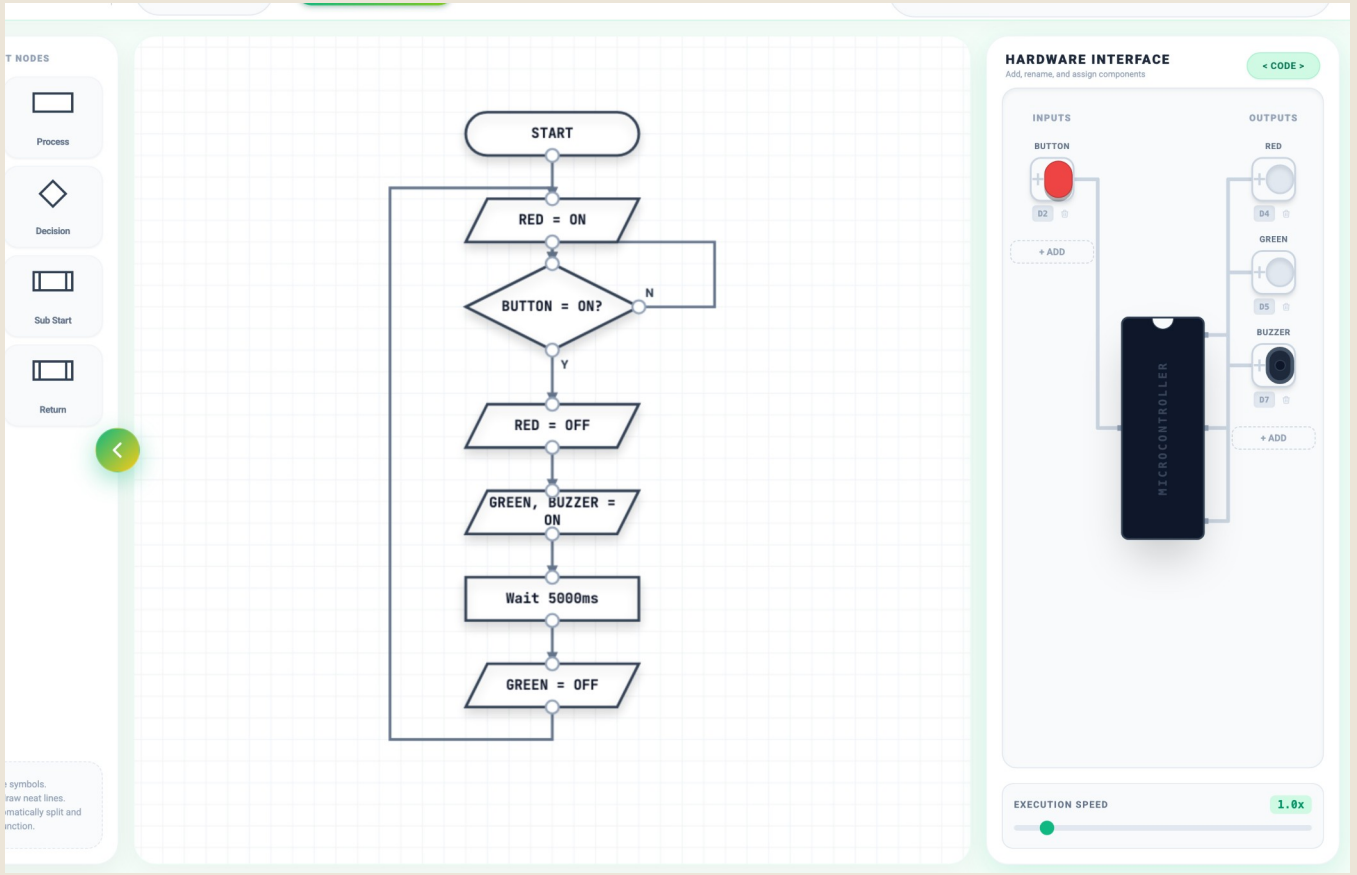
GREEN = OFF

## Missing

BUZZER = OFF

## So

the beeper never stops



# Say What You Saw

## THE MISTAKE

- It didn't work.
- Fixed it.
- Works now.

## THE CORRECT ANSWER

- Beeper stayed on after 5 s.
- Add pin 7 off after pin 5 off.
- Re-test: beeper stops at 5 s.

# Final Version: Save It

## After the re-test

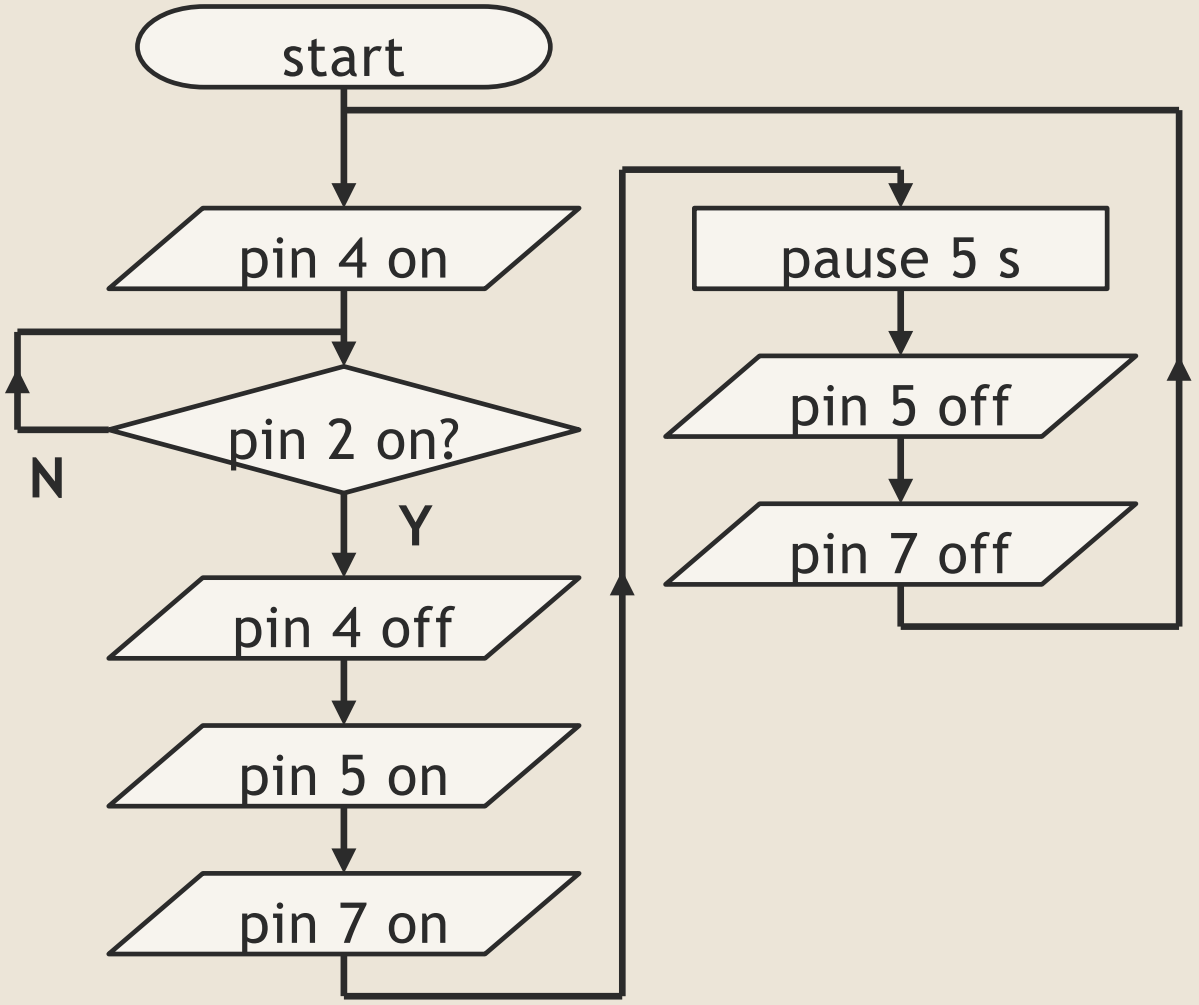
save the final version

## Give it

a new name:
crossing_FINAL

## It shows

every amendment, all tests passed



# Model Answer: Justify an Improvement (2 marks)

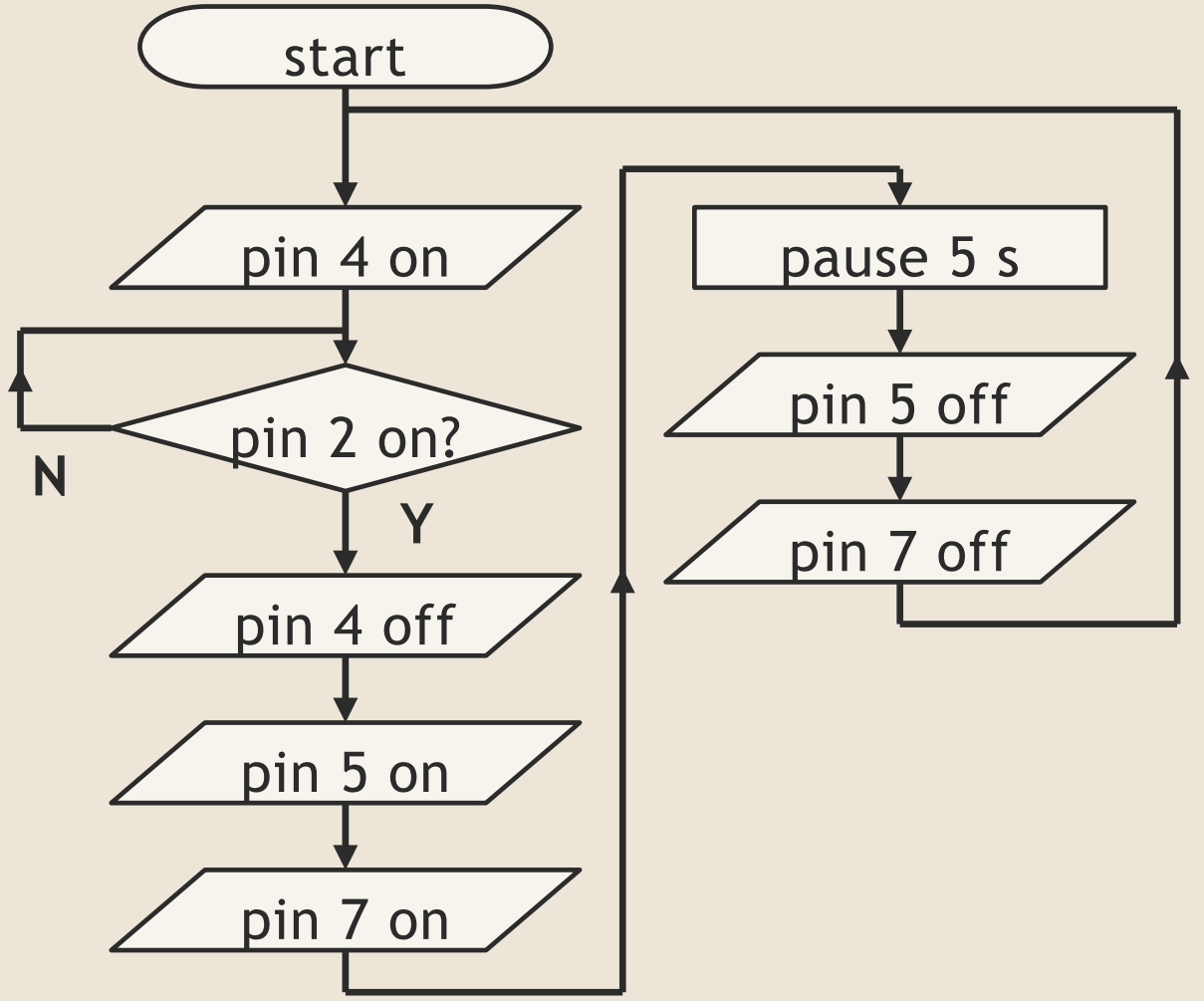
**Suggest** one **improvement. Justify** it. (2)

## Improvement (1 mark)

Make the beeper pulse on and off

## Justification (1 mark)

so blind pedestrians know when to cross.



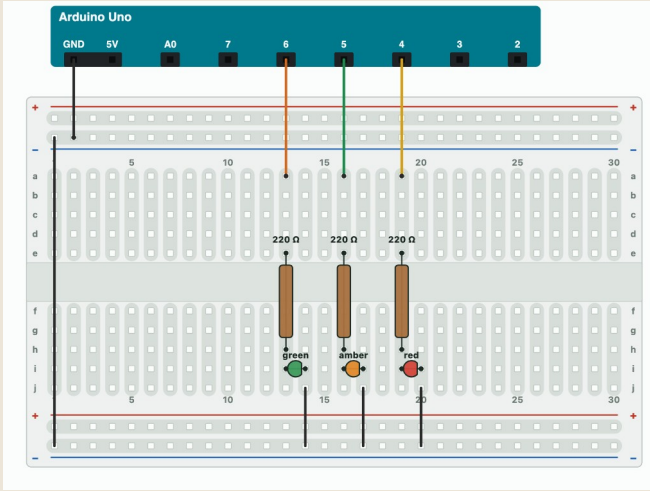
# Your Turn: Justify It




1. A pupil writes **'Make it better.' State** the **marks** it gets, and why.
2. A pupil writes **'Add a timer.' State** the **marks** it gets, and why.
3. The improvement is a **longer green man. Justify** it.

# Test It: Your Traffic Lights

# Practical sheet 3



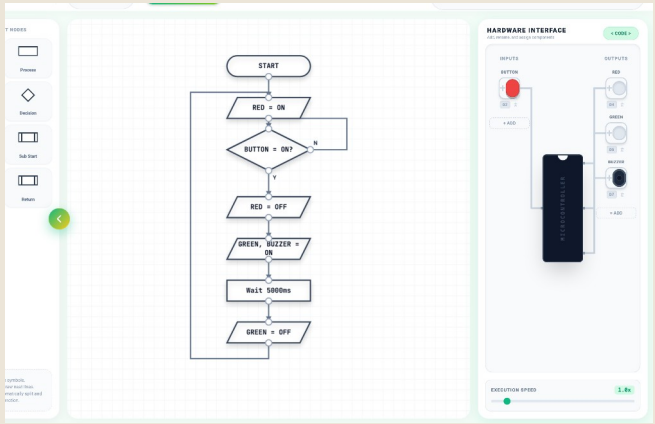

**Write 3 tests** for your **traffic lights.**
**Run** them. **Fill in every column.**

**SAFETY** USB power only. Unplug before you move a wire.

# Test It: The Faulty Crossing

# Practical sheet 3

**Test all 3 jobs**: NoSyntaxSim, then Yenka.
**Fill** the table. **Fix** it. **Re-test**.



# PRACTICE: Programming in Real Life

## Q7-8 (B4b-p23)

**Answer** Q7 and Q8 **on your own.**
Then **test** Q7 on your **Uno.**

7. Write Arduino C that sounds a buzzer on pin 8 for 3 s when a button on pin 2 is pressed, repeating forever. (4)

8. Convert this pseudocode to Arduino C: loop / high LED / pause 1 second / low LED / pause 1 second / end loop. (4)

# Answers: Programming in Real Life (B4b-p23)



| Question | Answer |
| --- | --- |
| **7 Write**: setup() | pinMode(2, INPUT); and pinMode(8, OUTPUT); |
| **7 Write**: loop() | if (digitalRead(2) == HIGH) { digitalWrite(8, HIGH); delay(3000); digitalWrite(8, LOW); } |
| **8 Convert**: setup() | pinMode(13, OUTPUT); (any pin, the same in all three lines) |
| **8 Convert**: loop() | digitalWrite(13, HIGH); delay(1000); digitalWrite(13, LOW); delay(1000); |

# Booklet Q7, Both Ways

## PSEUDOCODE

Flowchart

```
1  main:   if input2 is off then main
2          switch on 8
3          pause 3000
4          switch off 8
5          goto main
```

## ARDUINO C++

```
1   void setup() {
2     pinMode(2, INPUT);
3     pinMode(8, OUTPUT);
4   }
5   void loop() {
6     if (digitalRead(2) == HIGH) {
7       digitalWrite(8, HIGH);
8       delay(3000);
9       digitalWrite(8, LOW);
10    }
11  }
```

# Booklet Q8, Both Ways

## PSEUDOCODE

Flowchart

```
1 main:   switch on 13
2         pause 1000
3         switch off 13
4         pause 1000
5         goto main
```

## ARDUINO C++

```
1 void setup() {
2   pinMode(13, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(13, HIGH);
6   delay(1000);
7   digitalWrite(13, LOW);
8   delay(1000);
9 }
```

# Plenary: Part 3




1. **State** how many **tests** a program needs.
2. **Explain** why '**It didn't work**' is a **weak** result.
3. **State** what you must do **after a change**.

# Plenary: Part 3




4 **State** the **column** that records **what you saw**.

5 **State** how many **things** you **change** at once.

# Extension




1. Amber is **added** to the crossing. **State** where its **boxes** go.
2. **Describe** how to make the **beeper pulse**.
3. The **button** is **never read**. **State** the likely **fault**.

# Look back — by the end of this part…



1. I can **write a test** for **each job** a program does
2. I can **say what happened**, not just 'it failed'
3. I can **plan a change** and **test** it **again**

LOOKING AHEAD

# Booklet 4a: Logic and Digital Control

From programs to logic gates.

# Course Specification

## Digital electronic control systems

- using suitable commands (high, low, for…next, if…then, pause, end (or their equivalents)) to design programs to solve simple control problems, involving time delays and continuous and fixed loops

# Credits (1 of 2)

**The big question (An Arduino Uno):** Photo: JrawX, CC0, via Wikimedia Commons

**From Sim to Board (Uno photo):** Photo: JrawX, CC0, via Wikimedia Commons

**From Sim to Board (NoSyntaxSim):** NoSyntaxSim V2 screenshot. Simulator by Peter Strain, NoStrainSims (sites.google.com/view/nostrainsim), free and open to all.

**What to Collect (Uno photo):** Photo: JrawX, CC0, via Wikimedia Commons

**Breadboard and wiring drawings (all slides):** Breadboard and wiring drawings by R Stewart (Stripboard Builder engine).

**Watch Out (course specification):** © Qualifications Scotland (SQA). N5 Engineering Science course specification v2.0, p.9 (question paper).

**The Fault, in the Sim:** NoSyntaxSim V2 screenshot. Simulator by Peter Strain, NoStrainSims (sites.google.com/view/nostrainsim), free and open to all.

# Credits (2 of 2)

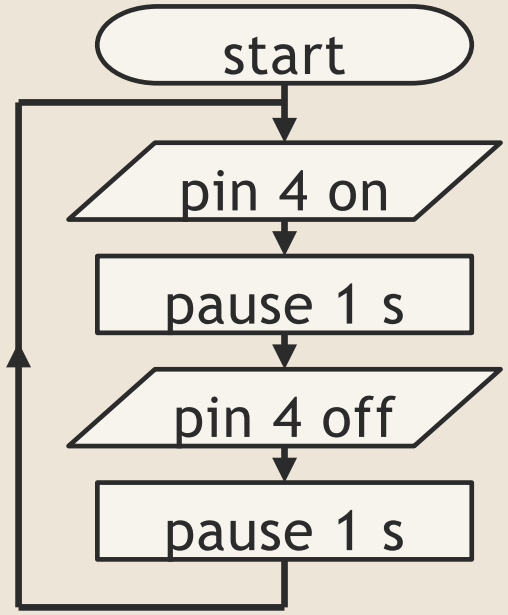
**Test It: The Faulty Crossing (NoSyntaxSim):** NoSyntaxSim V2 screenshot. Simulator by Peter Strain, NoStrainSims (sites.google.com/view/nostrainsim), free and open to all.

**PRACTICE: Programming in Real Life (booklet page):** Booklet 4b pages by R Stewart.

**Course Specification:** © Qualifications Scotland (SQA). Course spec v2.0, p.5

R Stewart - October 2026

# The Flowchart: Worked Example 1

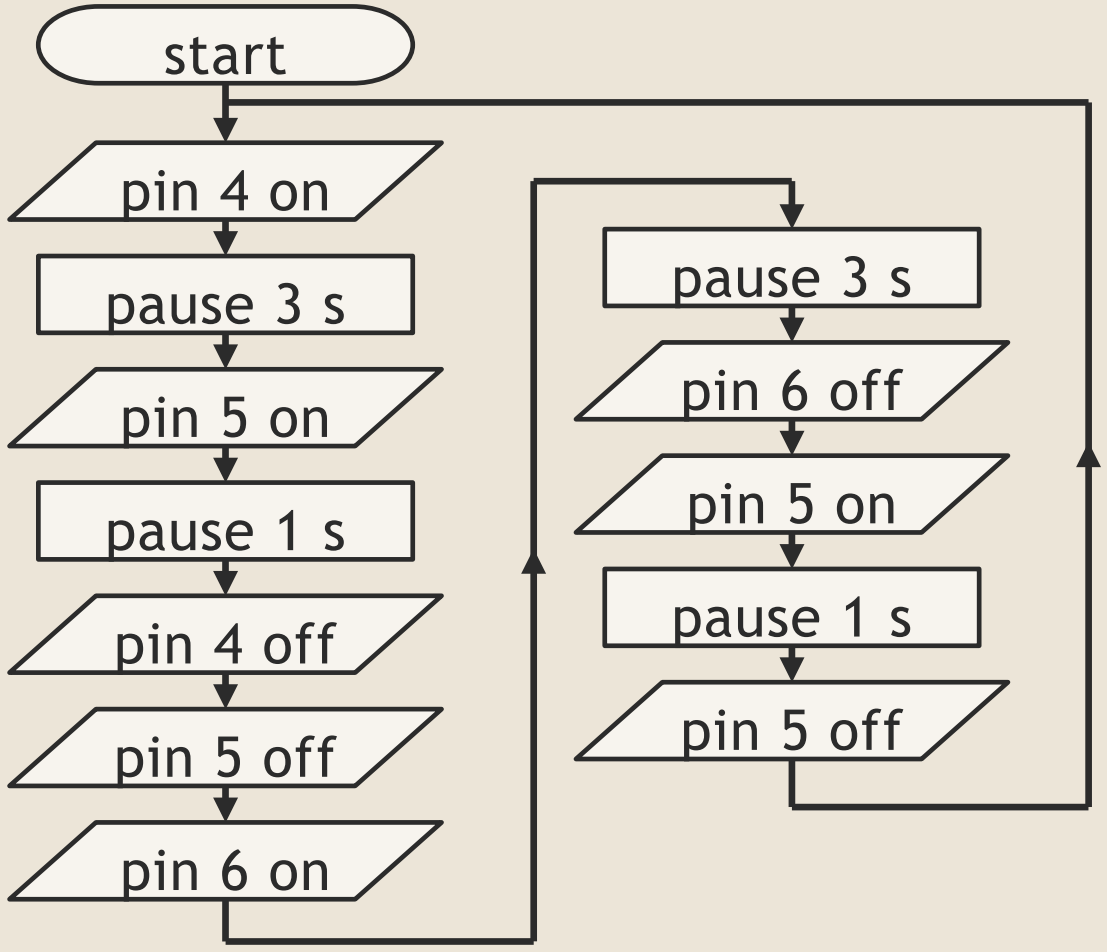



## ARDUINO C++

Back

```
1 void setup() {
2   pinMode(4, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(4, HIGH);
6   delay(1000);
7   digitalWrite(4, LOW);
8   delay(1000);
9 }
```

# The Flowchart: Worked Example 2

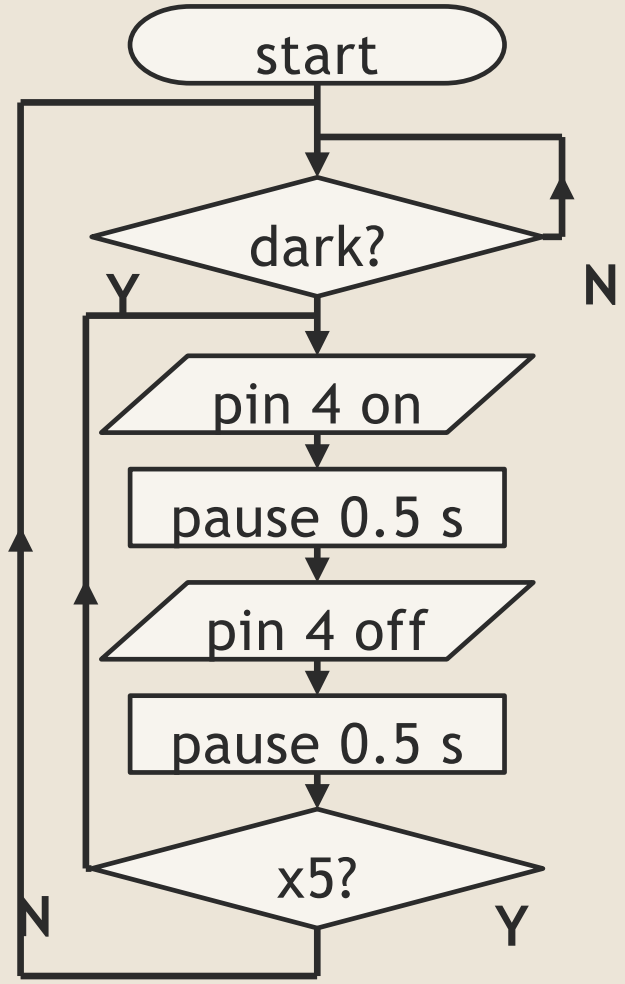



## ARDUINO C++

Back

```
1  void setup() {
2    pinMode(4, OUTPUT);
3    pinMode(5, OUTPUT);
4    pinMode(6, OUTPUT);
5  }
6  void loop() {
7    digitalWrite(4, HIGH);
8    delay(3000);
9    digitalWrite(5, HIGH);
10   delay(1000);
11   digitalWrite(4, LOW);
12   digitalWrite(5, LOW);
13   digitalWrite(6, HIGH);
14   delay(3000);
15   digitalWrite(6, LOW);
16   digitalWrite(5, HIGH);
17   delay(1000);
18   digitalWrite(5, LOW);
19 }
```

# The Flowchart: Worked Example 3

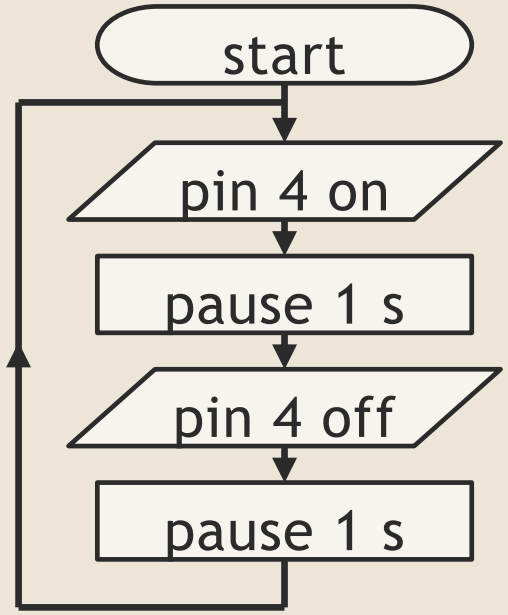



## ARDUINO C++

Back

```
1  void setup() {
2    pinMode(4, OUTPUT);
3  }
4  void loop() {
5    while (analogRead(A0) >= 300) { }
6    for (int count = 0; count < 5; count++) {
7      digitalWrite(4, HIGH);
8      delay(500);
9      digitalWrite(4, LOW);
10     delay(500);
11   }
12 }
```

# The Flowchart: Debug Build 1, Fixed

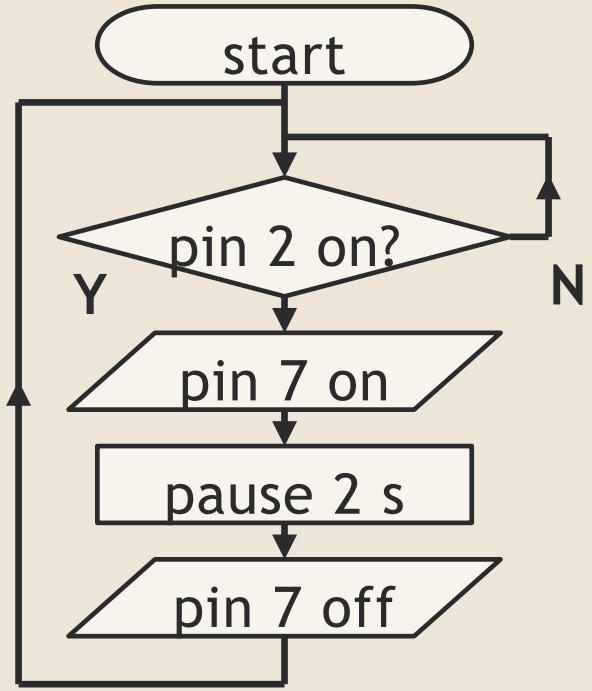



## ARDUINO C++

Back

```
1 void setup() {
2   pinMode(4, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(4, HIGH);
6   delay(1000);
7   digitalWrite(4, LOW);
8   delay(1000);
9 }
```

# The Flowchart: Debug Build 2, Fixed

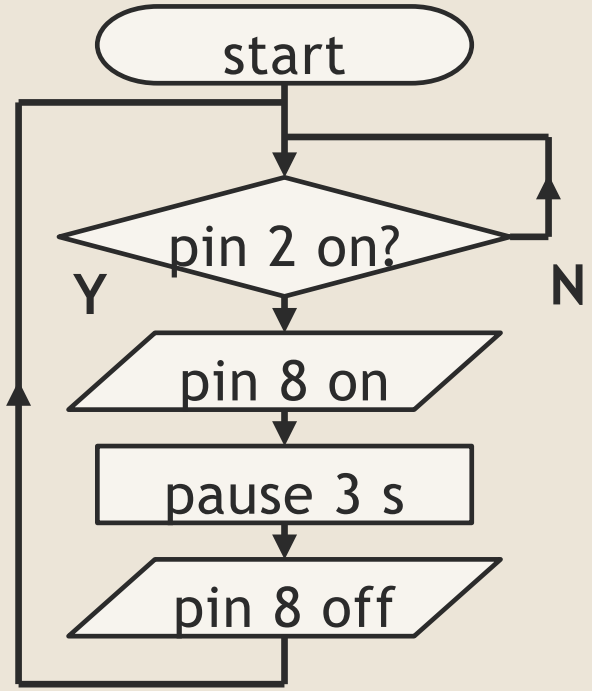



## ARDUINO C++

Back

```
1  void setup() {
2    pinMode(2, INPUT);
3    pinMode(7, OUTPUT);
4  }
5  void loop() {
6    if (digitalRead(2) == HIGH) {
7      digitalWrite(7, HIGH);
8      delay(2000);
9      digitalWrite(7, LOW);
10   }
11 }
```

# The Flowchart: Booklet Q7

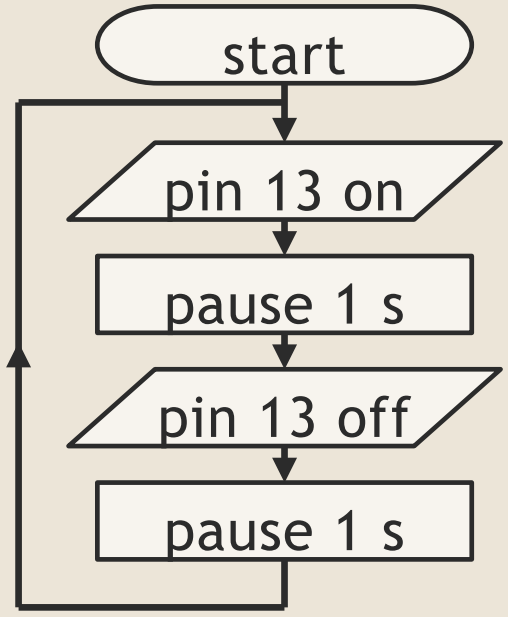



## ARDUINO C++

Back

```
1  void setup() {
2    pinMode(2, INPUT);
3    pinMode(8, OUTPUT);
4  }
5  void loop() {
6    if (digitalRead(2) == HIGH) {
7      digitalWrite(8, HIGH);
8      delay(3000);
9      digitalWrite(8, LOW);
10   }
11 }
```

# The Flowchart: Booklet Q8




## ARDUINO C++

Back

```
1 void setup() {
2   pinMode(13, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(13, HIGH);
6   delay(1000);
7   digitalWrite(13, LOW);
8   delay(1000);
9 }
```