The Victorian Curriculum
Digital Technologies

Programming in the Digital Technologies Curriculum (F-10)
VCAA Webinar – 15 March 2018

Daryl Croke – VCAA Specialist Teacher (Mount Ridley P-12 College)
Richard Fox – VCAA Specialist Teacher (Diamond Valley College)

Copyright

Victorian Curriculum
Foundation–10

VICTORIAN CURRICULUM
AND ASSESSMENT AUTHORITY

VICTORIA
State
Government

VCAA Professional Learning Support

To find online webinars or face-to-face sessions in your area:

http://www.vcaa.vic.edu.au/Pages/foundation10/viccurriculum/viccur-proflearn-specialists.aspx

To request a session for your local network:

http://www.vcaa.vic.edu.au/Pages/foundation10/viccurriculum/viccur-proflearn-specialists.aspx#request

Introduction

Daryl Croke, Mount Ridley P-12 College, Craigieburn

- Year 5/6 Digital Technologies
- Year 7 Robotics
- Year 8 Digital Technologies

Richard Fox, Diamond Valley College, Diamond Creek

- Year 7 Robotics
- Year 9/10 Electronics

Who’s with us today?

Are you participating on your own, or with colleagues?

What sector are you from?

What year level/s are you teaching?

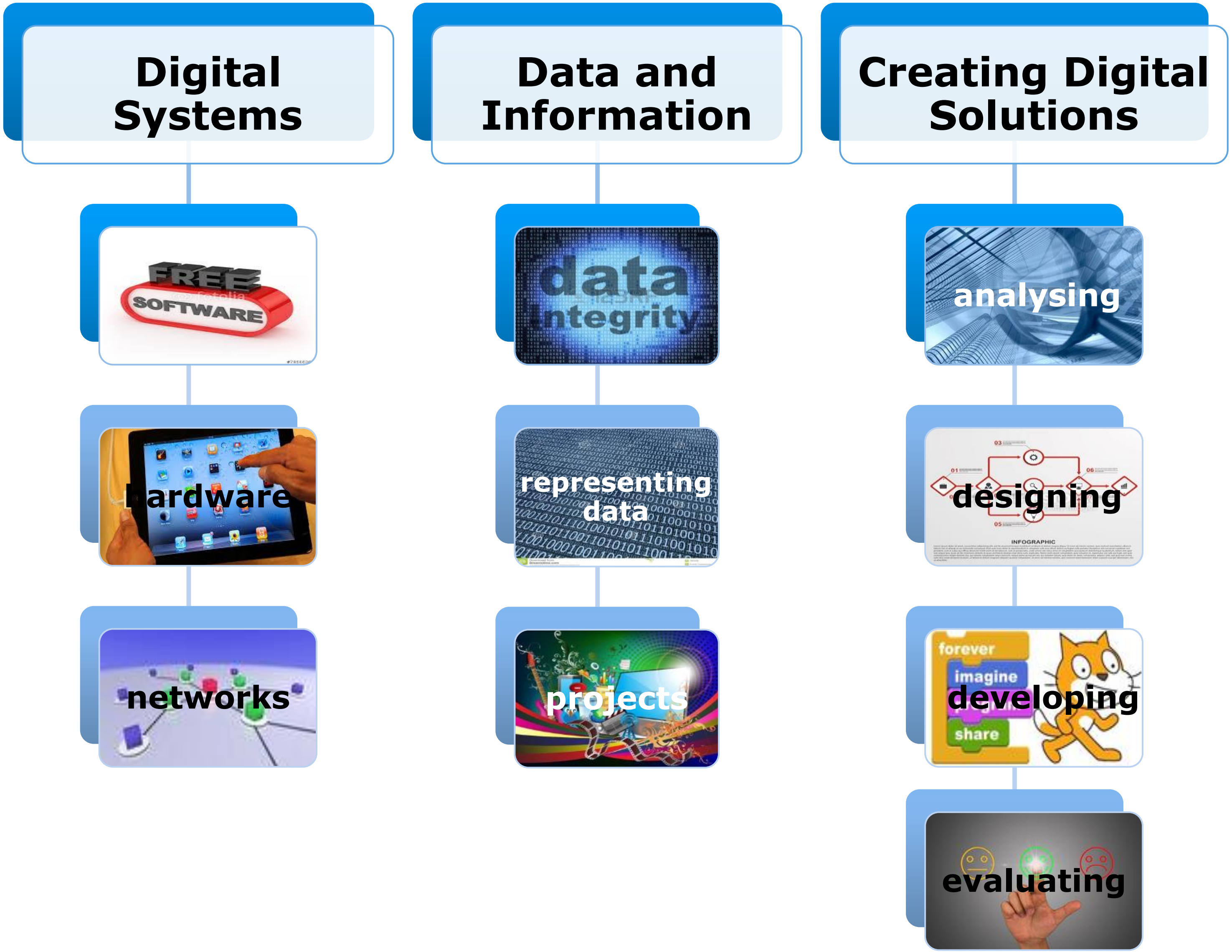
Is this your first year of teaching Digital Technologies?

Programming in the Digital Technologies Curriculum

Agenda

- Overview of the Creating Digital Solutions strand
- Ways of Thinking
- From algorithms to programming languages
- A look at some common programming environments (visual, general-purpose and object-oriented)

Overview of Creating Digital Solutions Strand



- **Digital Systems**
  - software
  - hardware
  - networks
- **Data and Information**
  - data integrity
  - representing data
  - projects
- **Creating Digital Solutions**
  - analysing
  - designing
  - developing
  - evaluating

Image credit: Paula Christophersen

Victorian Curriculum Foundation–10

VICTORIAN CURRICULUM AND ASSESSMENT AUTHORITY

VICTORIA State Government

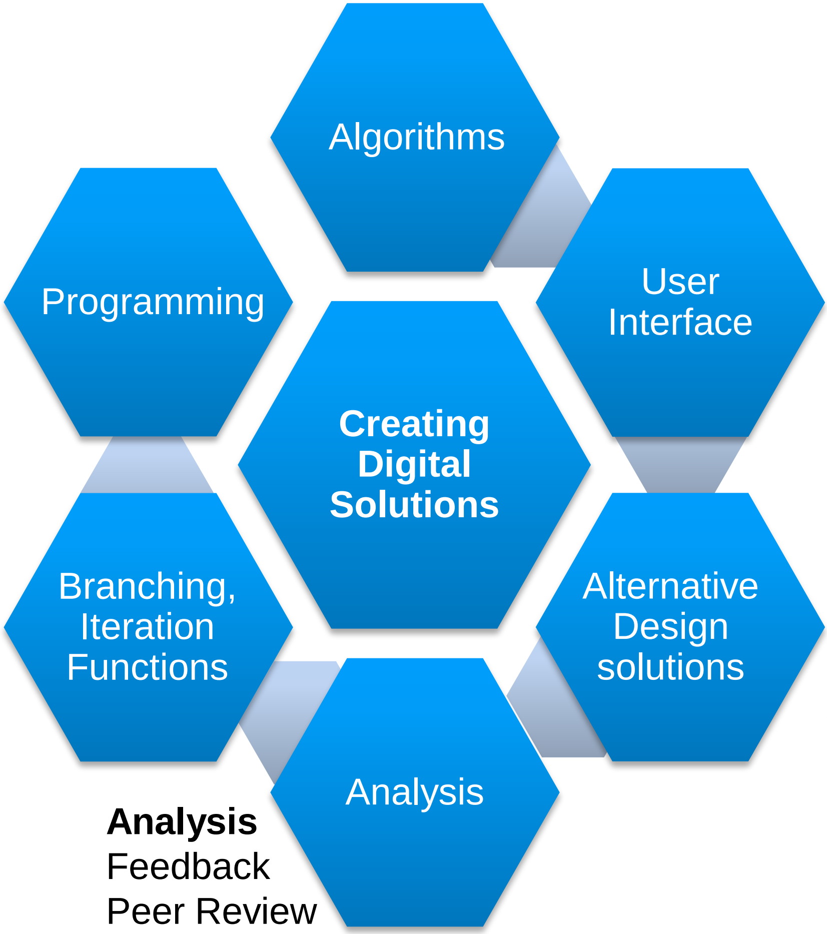
Curriculum is a continuum

Creating Digital Solutions	
Content Descriptions	
Levels 3 and 4	Develop **simple solutions** as visual programs
Levels 5 and 6	Develop digital solutions as **simple visual programs**
Levels 7 and 8	Develop and modify programs with user interfaces involving **branching**, **iteration** and **functions** using a **general-purpose programming language**
Levels 9 and 10	Develop modular programs, applying selected algorithms and **data structures** including using an **object-oriented programming language**

Creating Digital Solutions

Programming:
Visual
General-purpose
Object-oriented

Program Structure:
Branching
Iteration
Functions
Data Structures
Methods
Objects


Algorithms
Programming
User Interface
Creating Digital Solutions
Branching, Iteration Functions
Alternative Design solutions
Analysis

Algorithms
Series of instructions
Recipes
Procedures

User Interface
Requires a user to interact with the digital solution.

Alternative Design Solutions
Making modifications to current digital solutions

Analysis
Feedback
Peer Review
User Analysis
Reflective Process

Creating Digital Solutions

Explores processes and skills by which students create **digital solutions**

Four stages:

- Analysing
- Designing
- Developing
- Evaluating

Problem Solving Methodology

Creating Digital Solutions requires:

- Skills in using digital systems
- Different ways of thinking (computational, design and systems thinking)

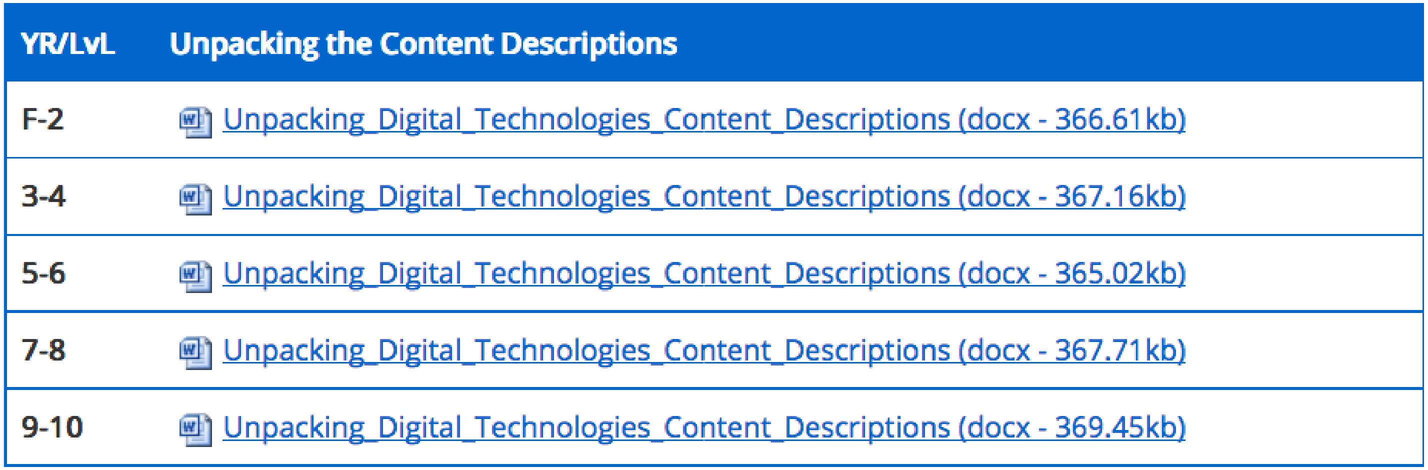
Links to other curriculum areas:

Mathematics, The Arts, Design and Technologies.

Teaching resources

The VCAA have some model lesson activities and sequences online for teachers to use

http://www.vcaa.vic.edu.au/Pages/foundation10/viccurriculum/digitech/teachresources.aspx



YR/LvL	Unpacking the Content Descriptions
F-2	Unpacking_Digital_Technologies_Content_Descriptions (docx - 366.61kb)
3-4	Unpacking_Digital_Technologies_Content_Descriptions (docx - 367.16kb)
5-6	Unpacking_Digital_Technologies_Content_Descriptions (docx - 365.02kb)
7-8	Unpacking_Digital_Technologies_Content_Descriptions (docx - 367.71kb)
9-10	Unpacking_Digital_Technologies_Content_Descriptions (docx - 369.45kb)

7-8 Sample content

Strand	Creating Digital Solutions
Content Description Develop and modify programs with user interfaces involving branching, iteration and functions using a general-purpose programming language	
Suggested Focus • overview of basic control structures used in general-purpose programming (sequence, branching and iteration) • introducing: • variables and data types • methods and data structures • procedures and functions that return a value • solving simple problems through the use of a general-purpose programming language • using testing tables and test data	

7-8 Sample content

Sample Activities

- transforming simple algorithms into programs using a nominated general-purpose programming language
- analysing more complex programs and identifying the variables used and their data types
- creating programs that incorporate all three control structures (sequence, branching and iteration)
- using functions that return values in a program
- modifying programs with simple data structures such as lists or arrays
- modifying supplied programs and predicting the expected output
- using various techniques to test the expected output of a program, such as testing tables

Ways of thinking

Ways of Thinking

The Digital Technologies curriculum draws on these important ideas:

- Computational Thinking
- Design Thinking
- Systems Thinking

They are embedded in the curriculum but not explicitly stated!

Computational Thinking

"A problem-solving method … that can be implemented by digital systems, such as **organising data** logically, **breaking down problems** into components, and the design and use of **algorithms**, patterns and models."

VCAA Digital Technologies Glossary

Design Thinking

"... understanding design problems and opportunities, **visualising** and generating creative and innovative ideas, and **analysing** and **evaluating** those ideas that best meet the **criteria** for success and planning. Designing stems from the notion that current products, processes, systems or services are either unsuitable for our needs or can be improved."

VCAA Digital Technologies Glossary

Systems Thinking

"… the identification and solving of problems where parts and components of a system, their **interactions** and **interrelationships** are analysed individually to see how they influence the functioning of the whole system. This approach enables students to **understand systems** and work with complexity, uncertainty and risk."

VCAA Digital Technologies Glossary

Computational Thinking

When we are creating algorithms and turning them into program code, we are making use of **Computational Thinking**.

From Algorithms to Programming Languages

Solving problems

All problems can be solved with algorithms

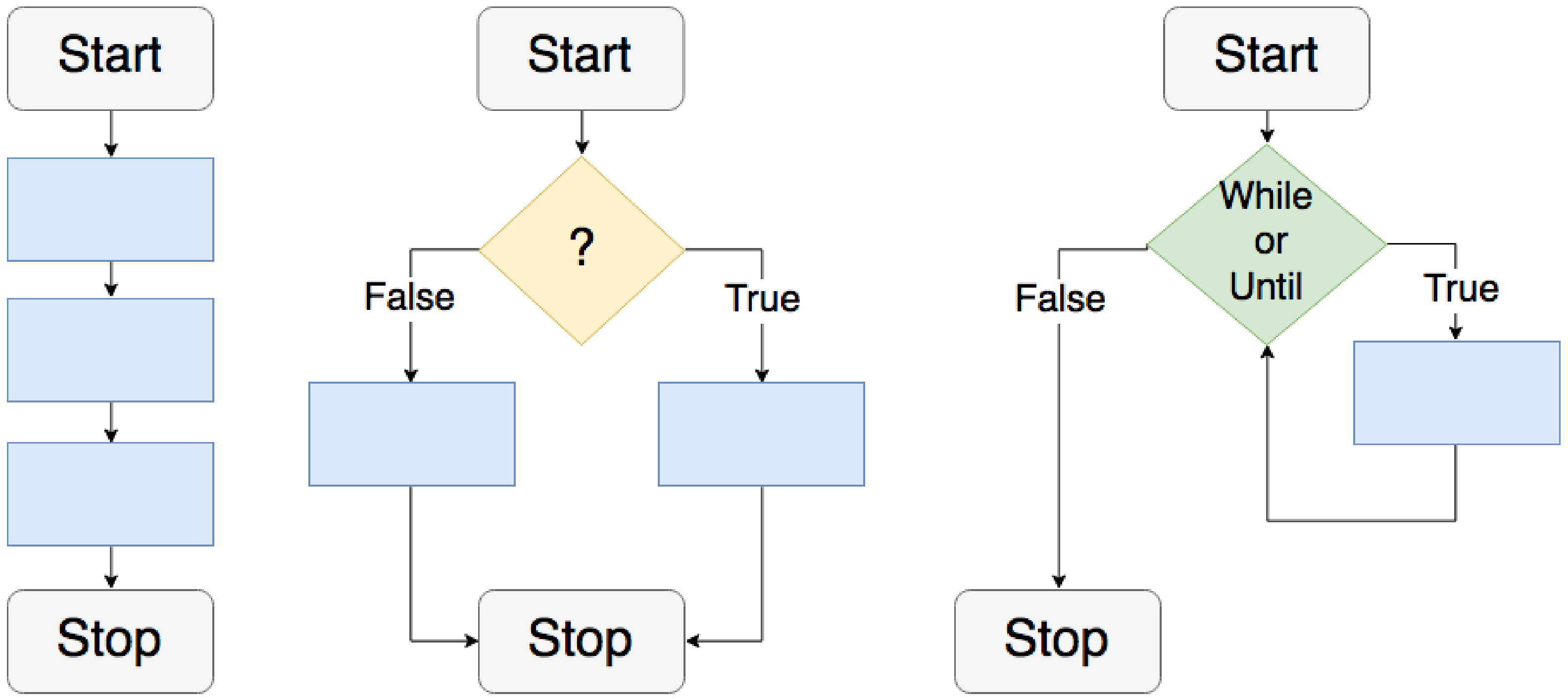
Algorithm: “A description of the steps and decisions required to solve a problem.”

VCAA Digital Technologies Glossary

*Algorithms are the **thinking** behind programs, poor thinking poor program*

Control structures

All problems can be solved using three control structures

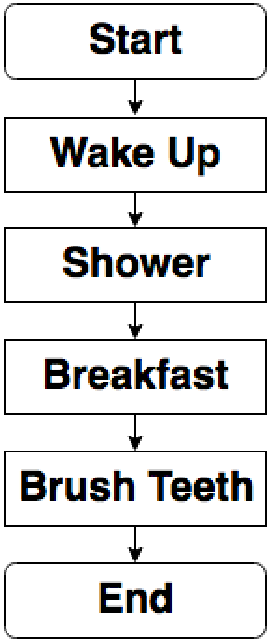


Sequence

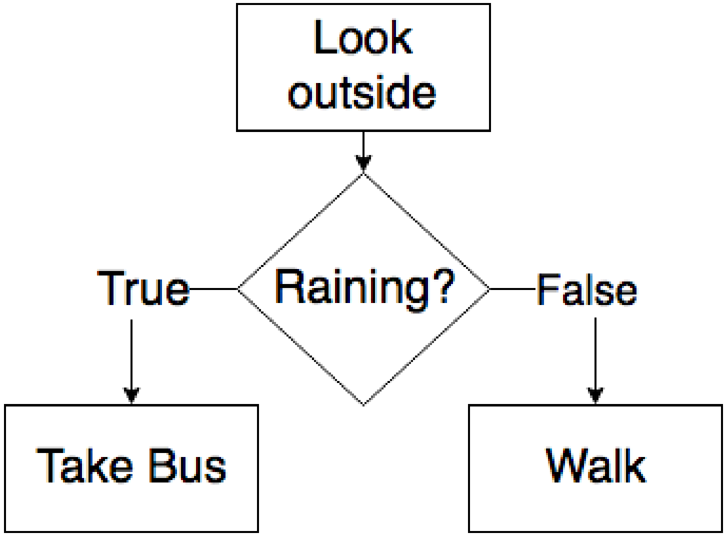
Branching

Iteration

Sequence (step by step)

Control Structure | Sequence	
Diagrammatically	**English**
 Start Wake Up Shower Breakfast Brush Teeth End	Start Wake Up Shower Eat Breakfast Brush Teeth Stop

Branching (selection)

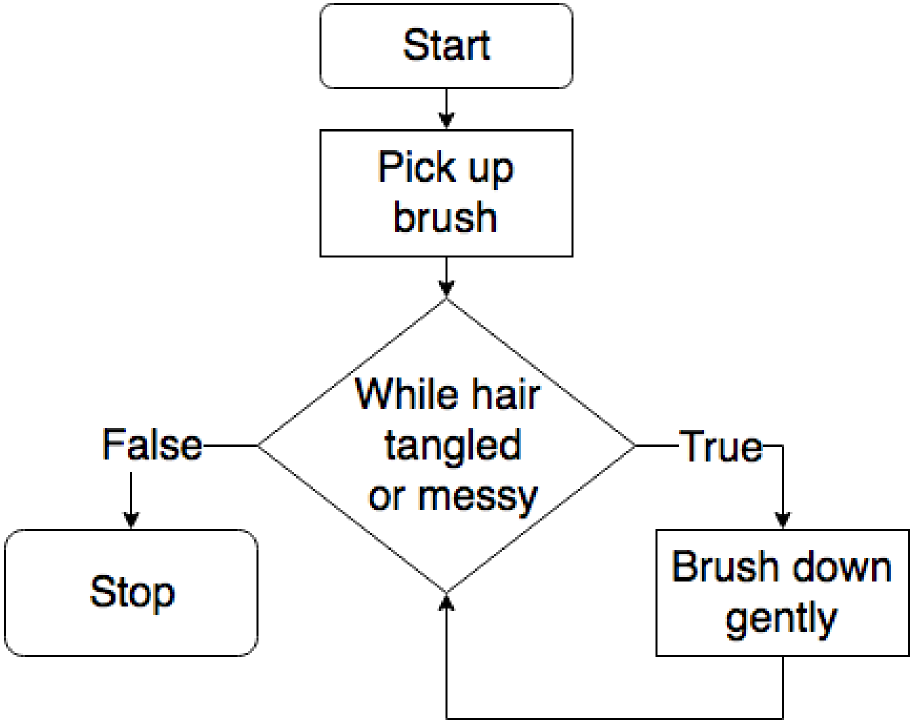
<table>
<tr><th colspan="2">Control Structure | Branching</th></tr>
<tr><th>Diagrammatically</th><th>English</th></tr>
<tr><td>

Look outside
Raining?
True
False
Take Bus
Walk

</td><td>Look outside
If raining
    Take the bus
Else
    Walk to school</td></tr>
</table>

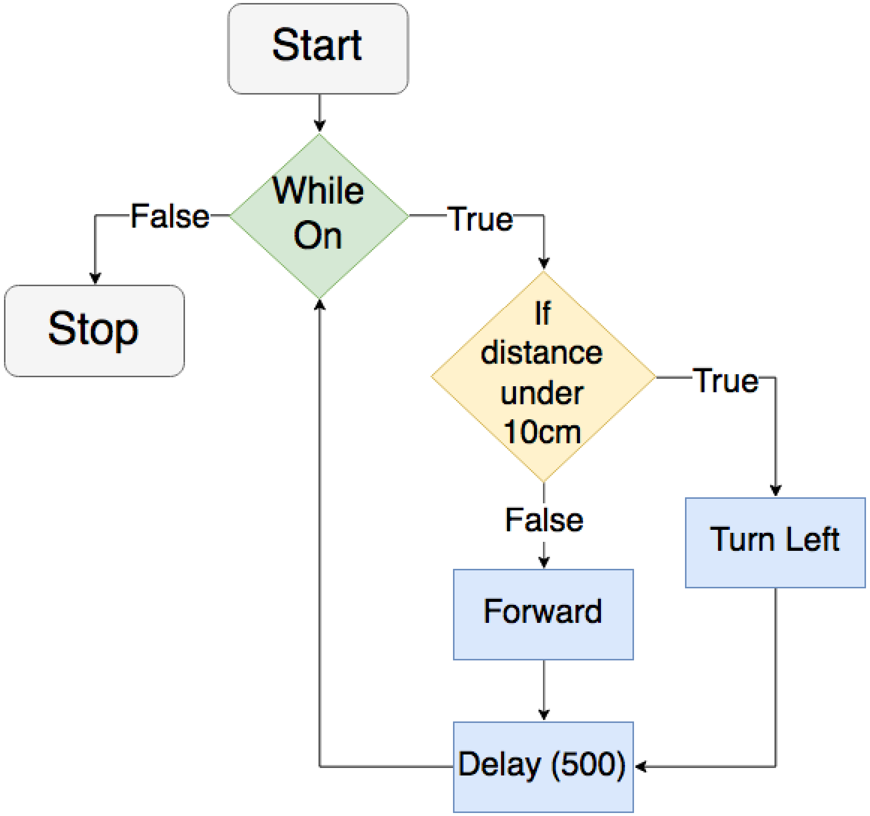
Iteration (repeating, looping)

Control Structure | Iteration (repeating, looping)
Diagrammatically



Combining structures

Combination (sequence, iteration, branching)



Starting to program

Questions

Who’s teaching programming now?

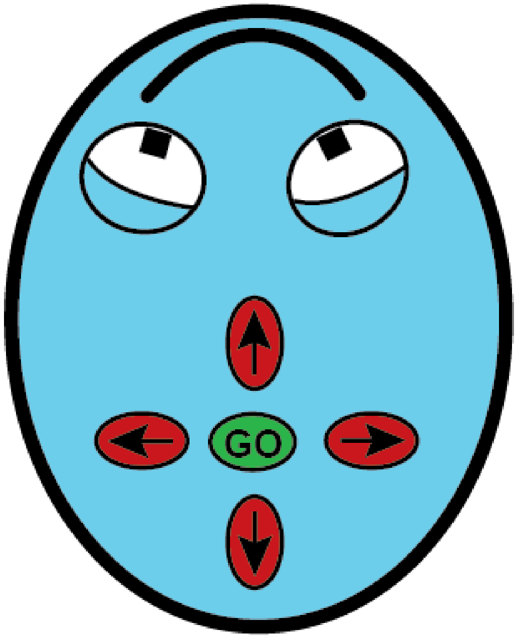
What are you using?

How is it going?

Programming F-2 level

At F-2 levels we would only concentrate on sequence (step by step) instructions

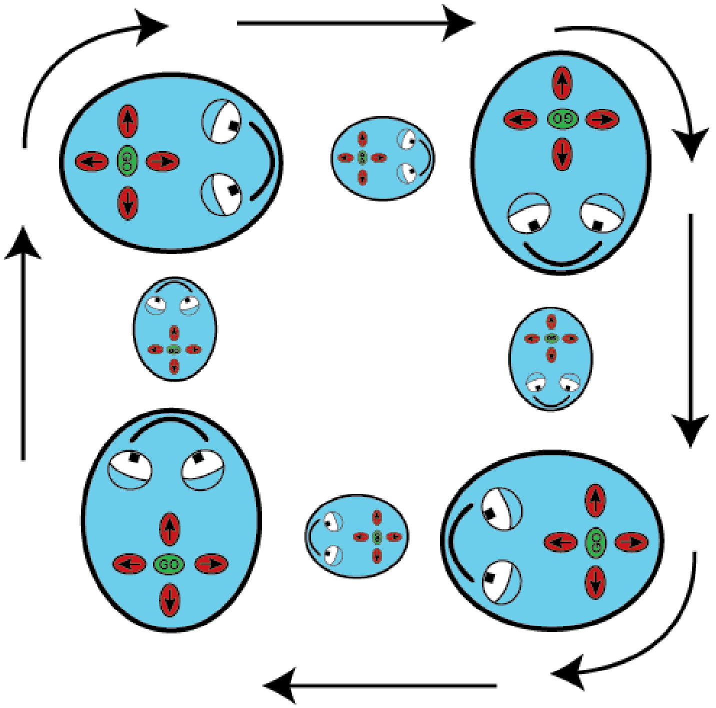
A good example would be programming a Bee-Bot type robot



Level F to 2

Programming F-2 level

Problem: how to we program a square?



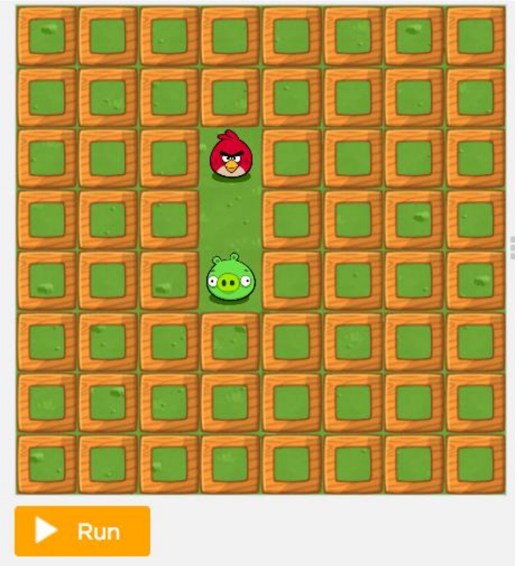
Bee-Bot Square Program	
Forward	⇧
Right	⇨
Forward	⇧
Right	⇨
Forward	⇧
Right	⇨
Forward	⇧
Right	⇨

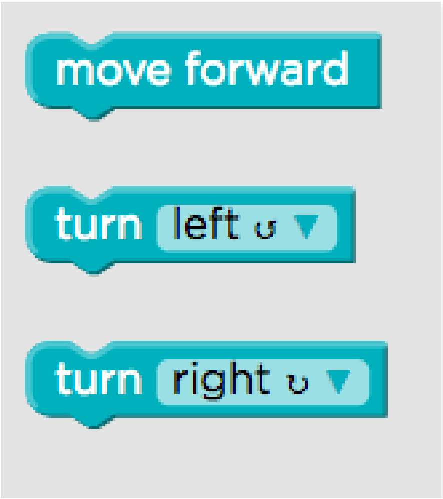
Level 3 and up

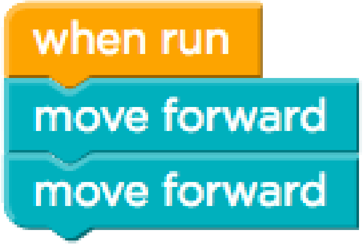
Programming 3-4

I start with Code.org “Classic Maze”: https://studio.code.org/hoc/1

What control structure is being used?




```
move forward

turn left ↺ ▼

turn right ↻ ▼
```

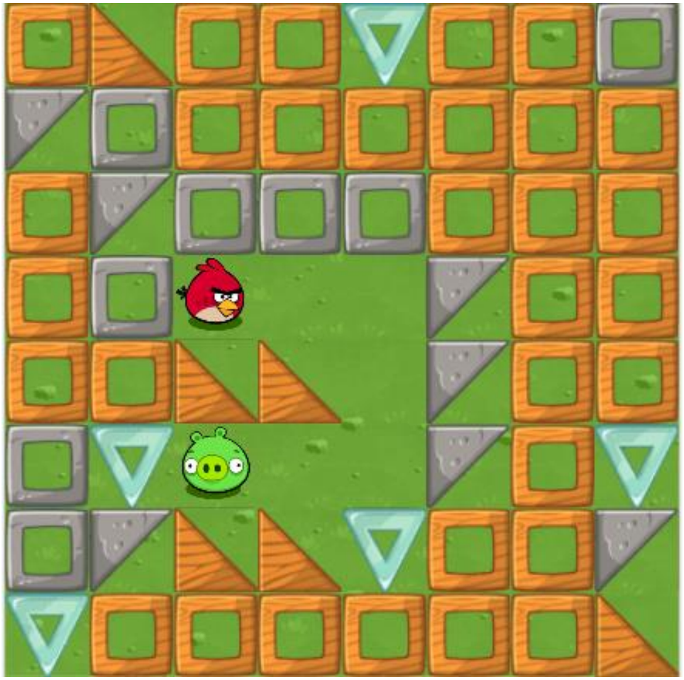

```
when run
move forward
move forward
```

Level 5 and up

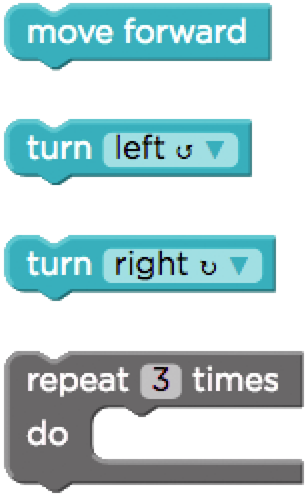
Links with Algorithms

Students get stuck on Problem 9

What is the solution to this problem?



Students can use the following Blocks



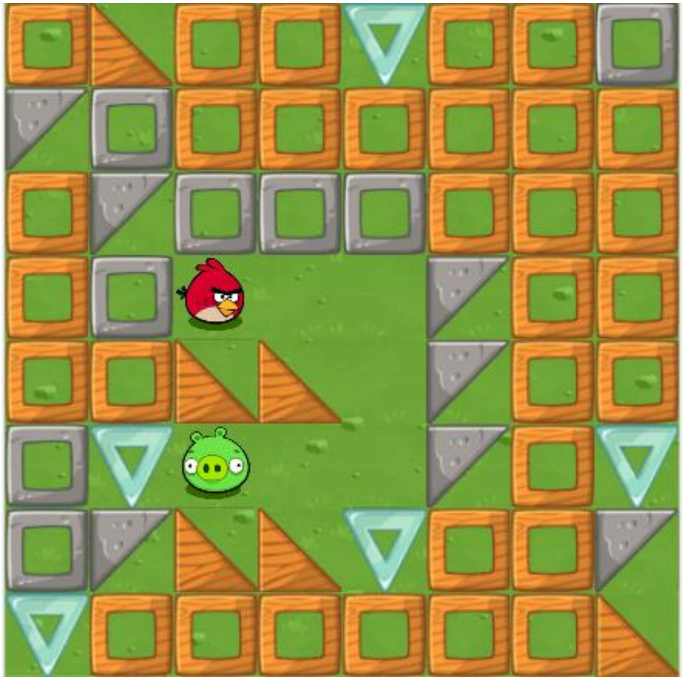
```
move forward
turn left ↺
turn right ↻
repeat 3 times
do
```

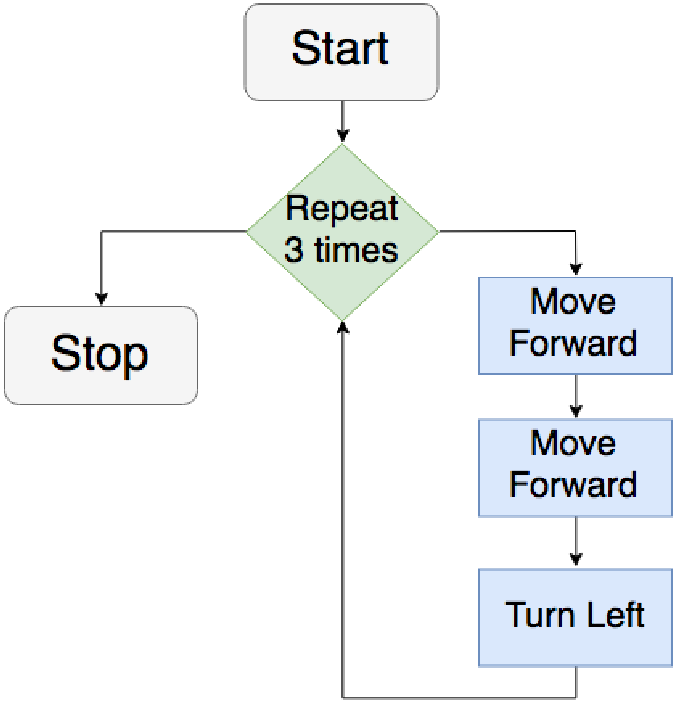
Level 5 and up

Links with Algorithms

Students get stuck on Problem 9

Many students don't realise that a repeating pattern can contain more than one action.

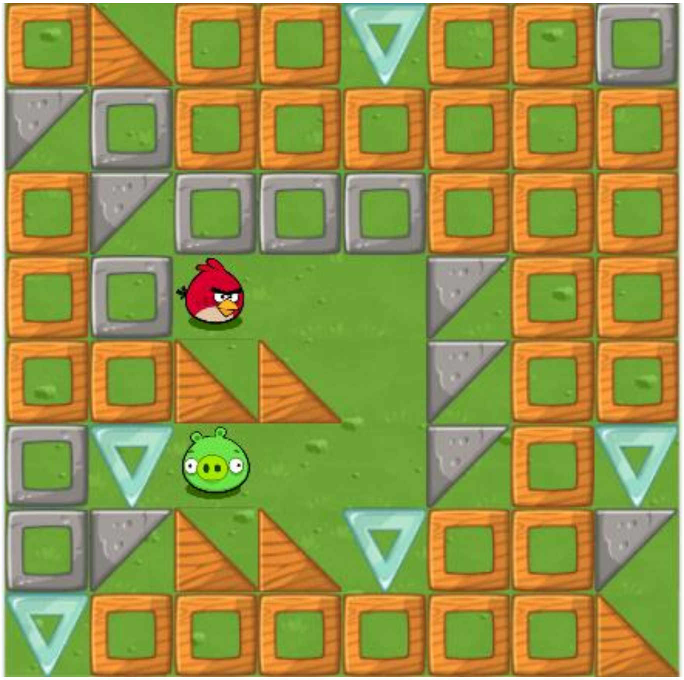


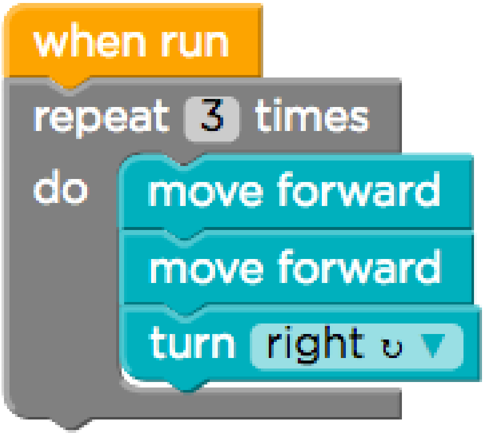


Level 5 and up

Solution

(move forward, move forward, turn right) x 3





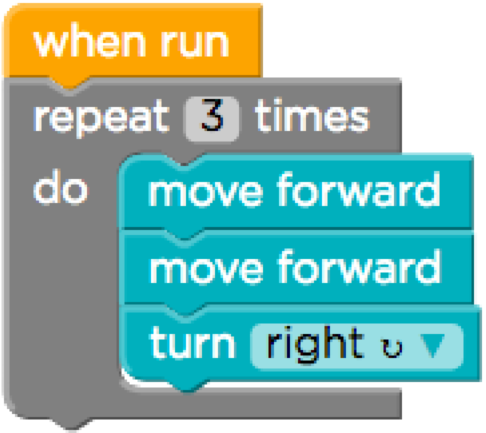
```
when run
repeat 3 times
do  move forward
    move forward
    turn right ↻
```

Level 5 and up

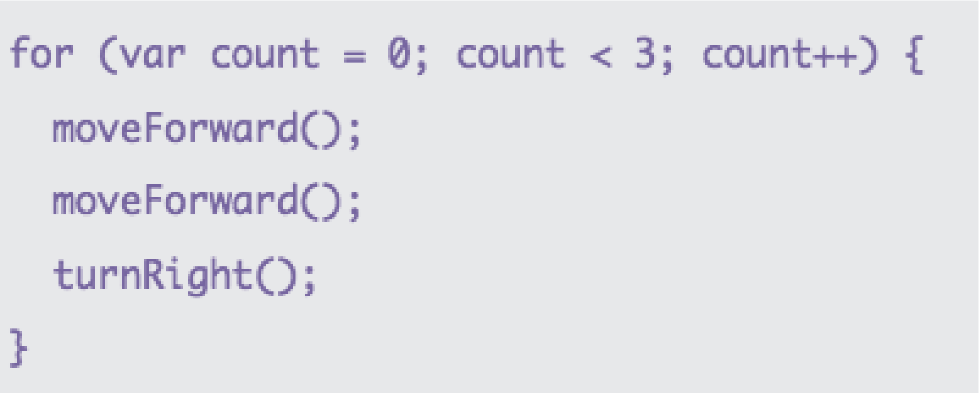
Blockly and Text-based code

Encourage students to look at the JavaScript code

Ask them to find examples of each control structure (home work task)



```
when run
repeat 3 times
do  move forward
    move forward
    turn right ↻
```

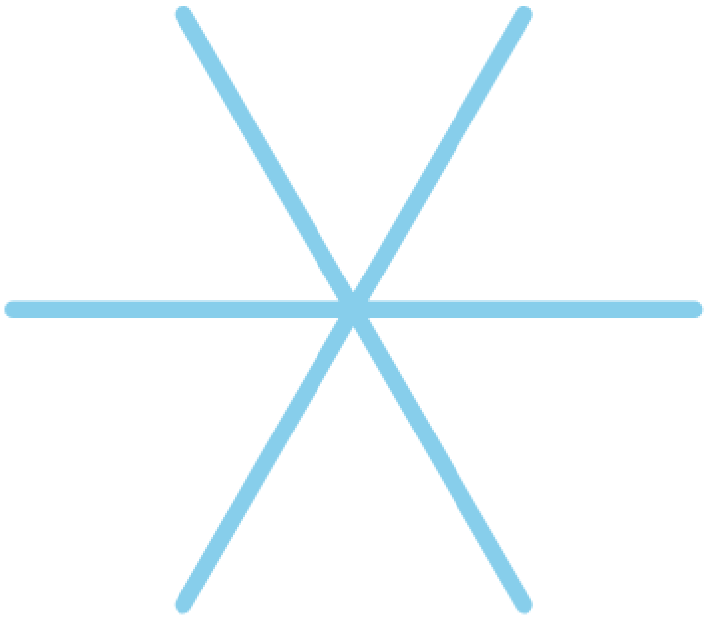


```
for (var count = 0; count < 3; count++) {
  moveForward();
  moveForward();
  turnRight();
}
```

Level 5 and up

Problem to Program

Problem: Draw a 6 pointed star



What is a possible algorithm to achieve this?

What ‘blocks’ or control structures do we need?

https://groklearning.com/learn/hoc-snowflake-blockly/

Victorian Curriculum
Foundation–10

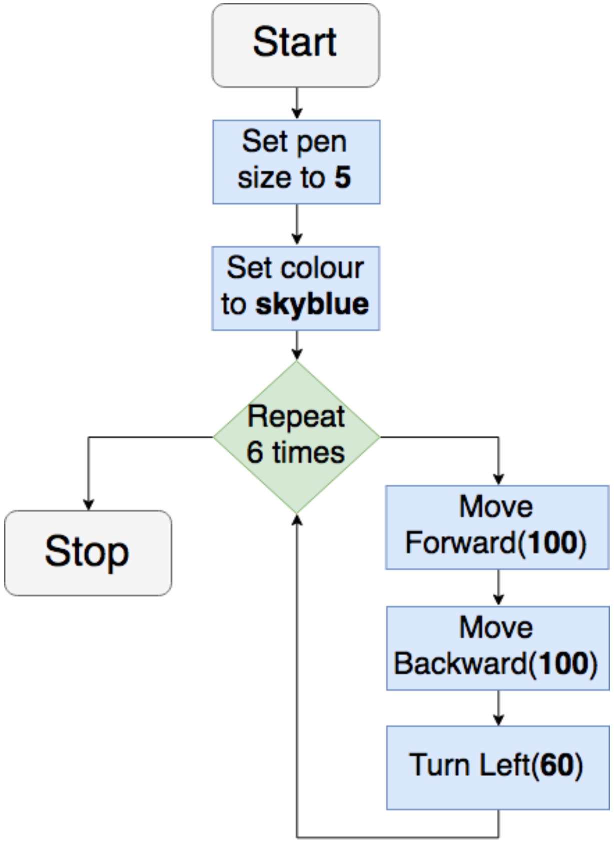
VICTORIAN CURRICULUM AND ASSESSMENT AUTHORITY

VICTORIA State Government

Level 7 and up

Problem to Program

Diagram Algorithm



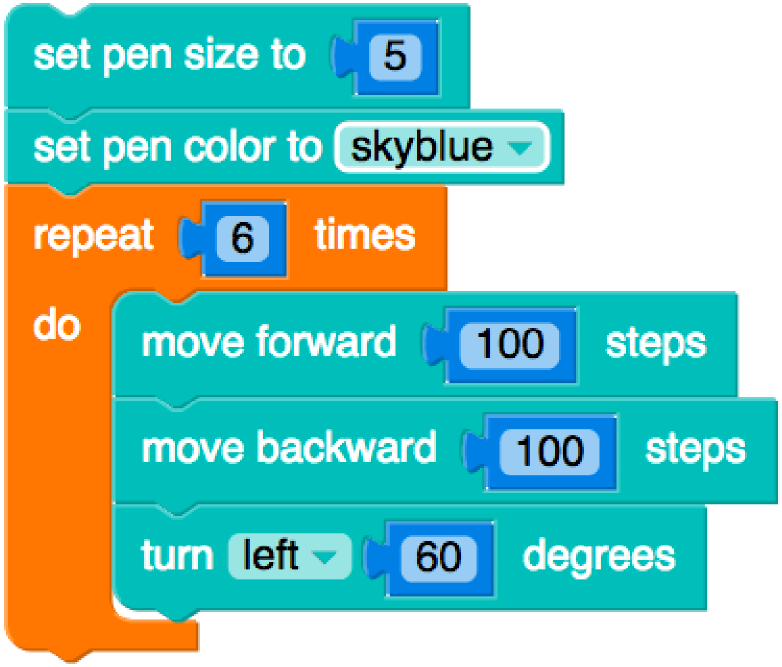
English Algorithm

```
Start
Set pen size to 5
Set pen colour to “skyblue”

Repeat 6 times
            Move Forward 100
            Move Backward 100
            Turn Left 60
End Repeat
End
```

Level 7 and up

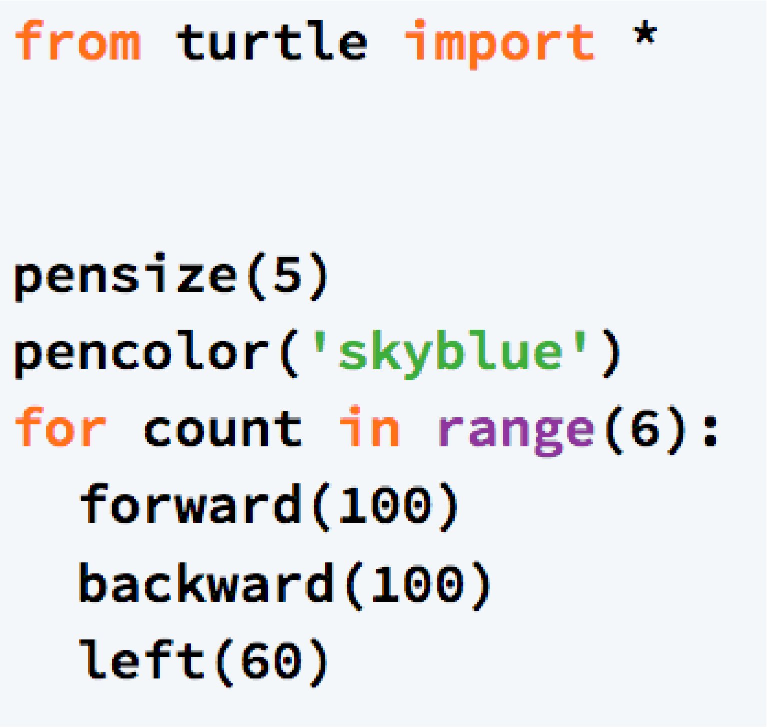
Problem to Program

Block Based Coding



```
set pen size to 5
set pen color to skyblue
repeat 6 times
do  move forward 100 steps
    move backward 100 steps
    turn left 60 degrees
```

Text-Based Coding



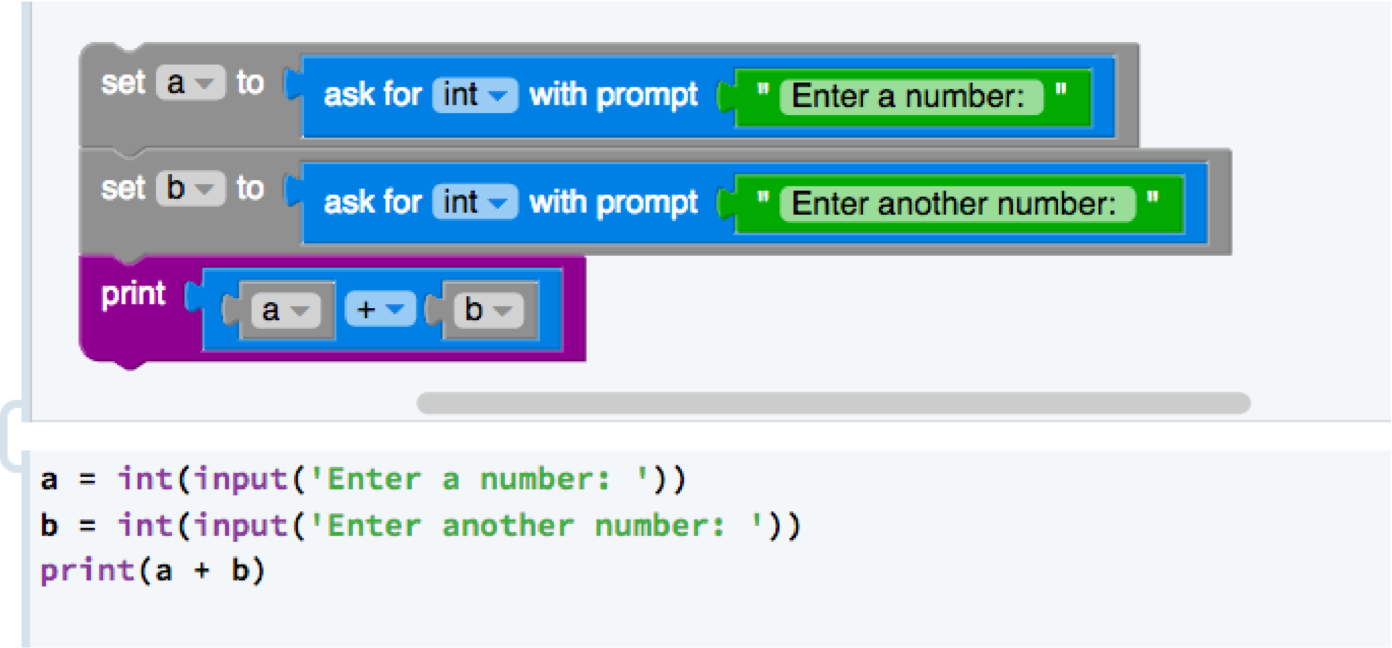
```
from turtle import *


pensize(5)
pencolor('skyblue')
for count in range(6):
  forward(100)
  backward(100)
  left(60)
```

https://groklearning.com/learn/hoc-snowflake-blockly/

Level 7 and up

User Input (Integers)

Toggling between blocks and text



set a to ask for int with prompt " Enter a number: "
set b to ask for int with prompt " Enter another number: "
print a + b

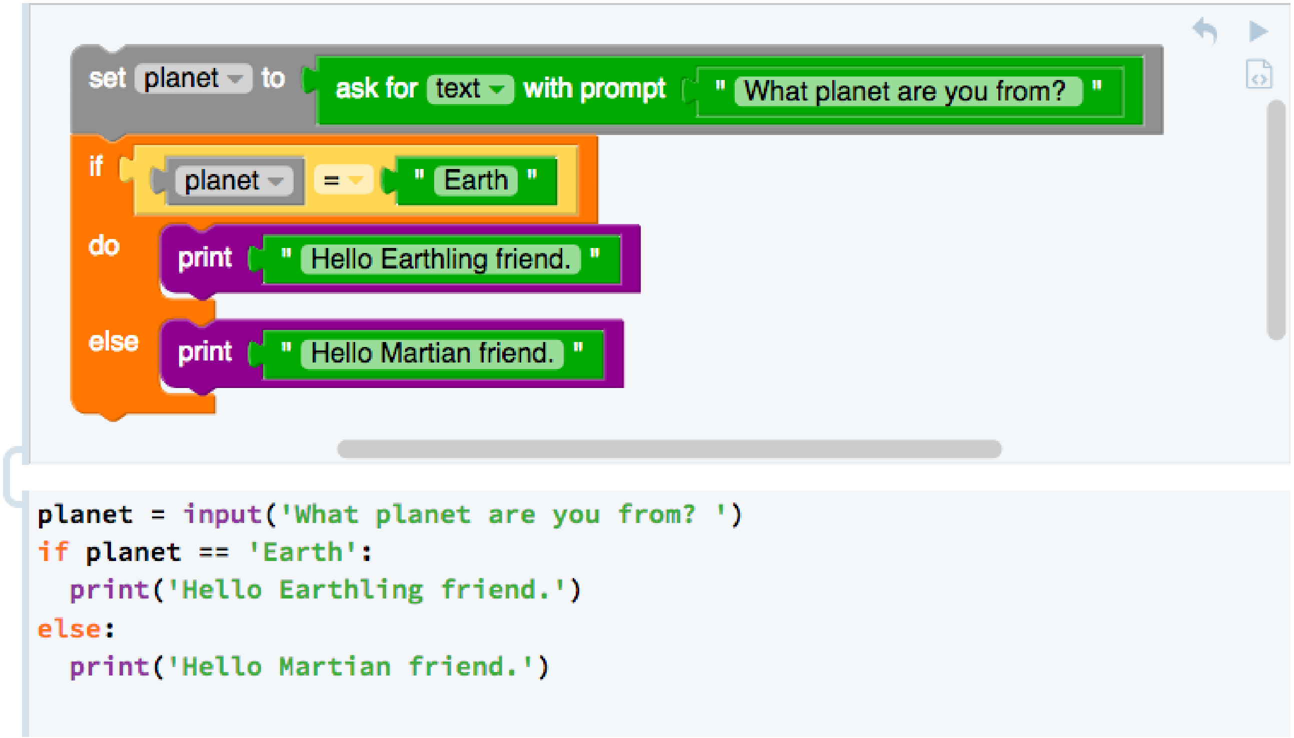
```
a = int(input('Enter a number: '))
b = int(input('Enter another number: '))
print(a + b)
```

https://groklearning.com/learn/hoc-space

Level 7 and up

User Input (strings)

Using Branching (if/else statements)


set planet to ask for text with prompt " What planet are you from? "
if planet = " Earth "
do print " Hello Earthling friend. "
else print " Hello Martian friend. "

```
planet = input('What planet are you from? ')
if planet == 'Earth':
  print('Hello Earthling friend.')
else:
  print('Hello Martian friend.')
```

https://groklearning.com/learn/hoc-space

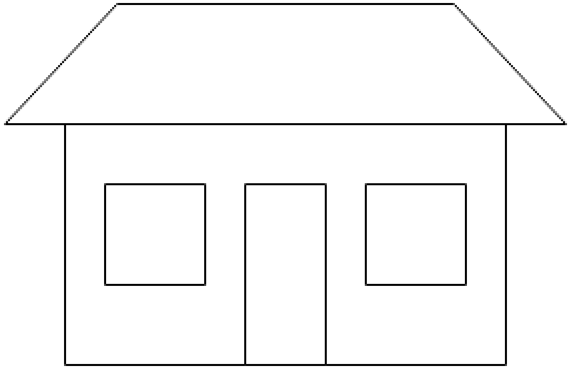
Level 7 and up

Functions

Functions are introduced at Level 7 and 8.

"A function is a sequence of instructions that we can define and reuse multiply times"

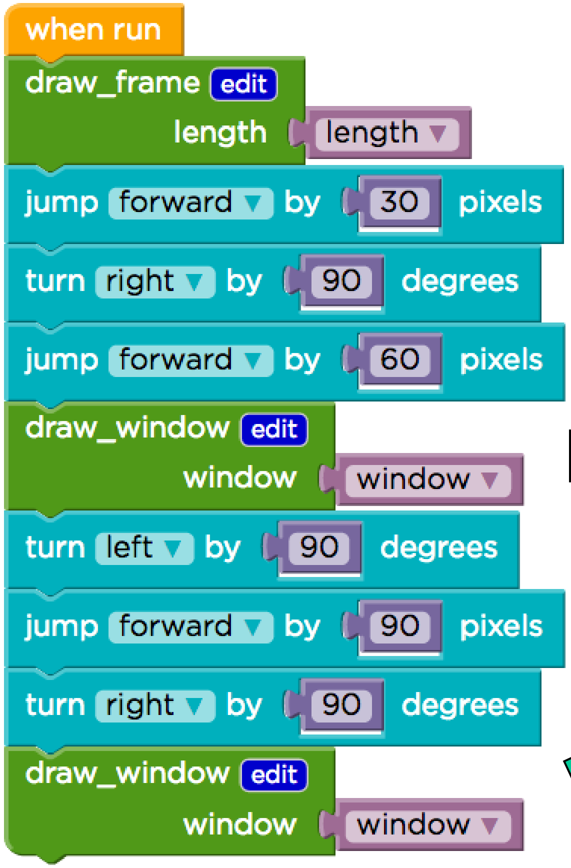
Example of use: We want to draw a house with 2 windows. Rather than creating 2 blocks of repeating code we would **create a function** to draw 1 window and **call the function** 2 times.



Level 7 and up

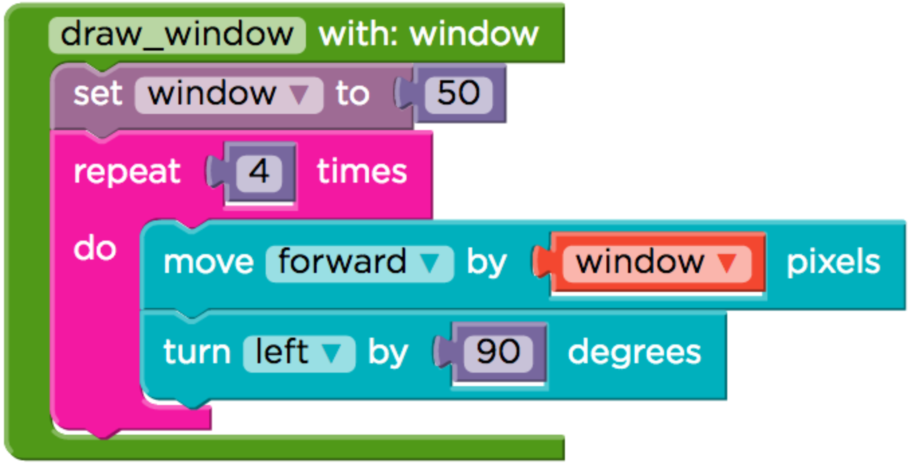
Functions

draw_window function is ***called*** in the main program



```
when run
draw_frame edit
    length length
jump forward by 30 pixels
turn right by 90 degrees
jump forward by 60 pixels
draw_window edit
    window window
turn left by 90 degrees
jump forward by 90 pixels
turn right by 90 degrees
draw_window edit
    window window
```

draw_window function is ***defined*** elsewhere

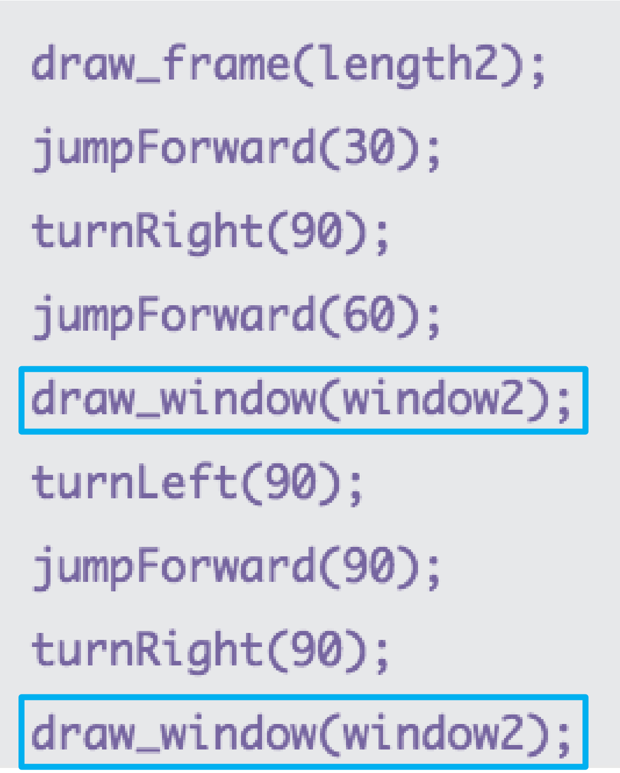


```
draw_window with: window
  set window to 50
  repeat 4 times
  do
      move forward by window pixels
      turn left by 90 degrees
```

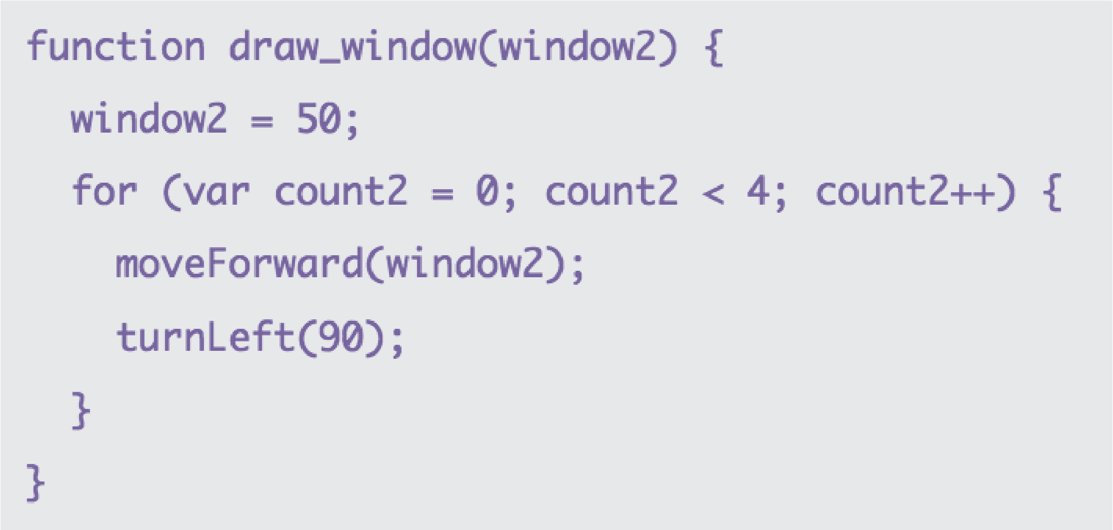
Level 7 and up

Functions

draw_window function is ***called*** in the main program



```
draw_frame(length2);
jumpForward(30);
turnRight(90);
jumpForward(60);
draw_window(window2);
turnLeft(90);
jumpForward(90);
turnRight(90);
draw_window(window2);
```



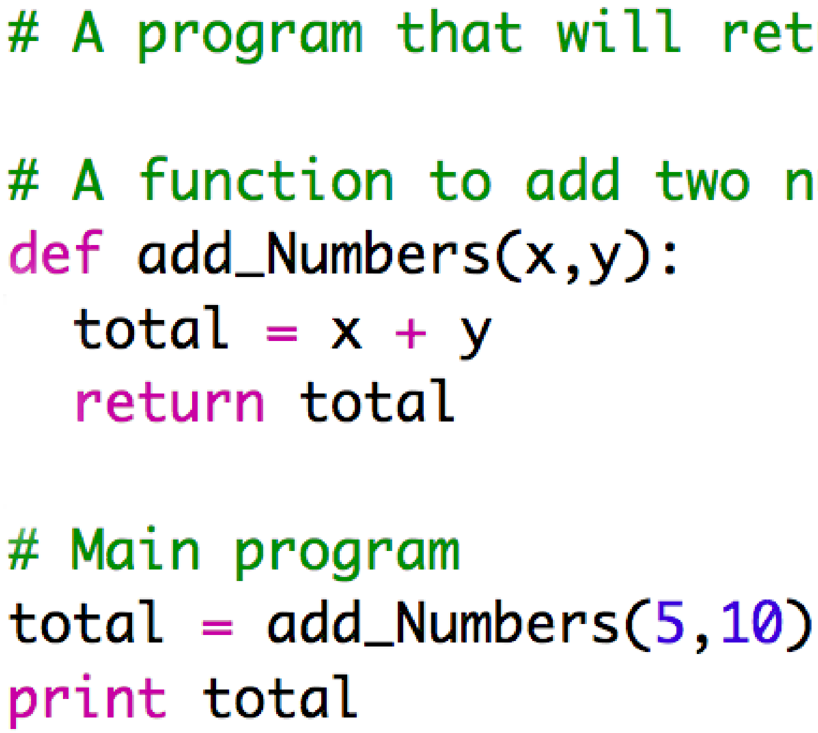
```
function draw_window(window2) {
  window2 = 50;
  for (var count2 = 0; count2 < 4; count2++) {
    moveForward(window2);
    turnLeft(90);
  }
}
```

functions are usually defined above the body of the main program

Level 7 and up

Functions

Real functions return a value, a simple Python example



```
# A program that will return a value

# A function to add two numbers
def add_Numbers(x,y):
  total = x + y
  return total

# Main program
total = add_Numbers(5,10)
print total
```

Level 9 and up

Object-oriented programming

At levels 9 and 10 students are introduced to Object-Oriented Programming

Using object-orientated programming allows the developer to simplify and reduce the lines of code

Similar to using functions, blocks of code are developed elsewhere and called upon in the main program

Object-oriented programming

Template and variation:

In object-oriented programming (OOP) we create a template called a **class**.

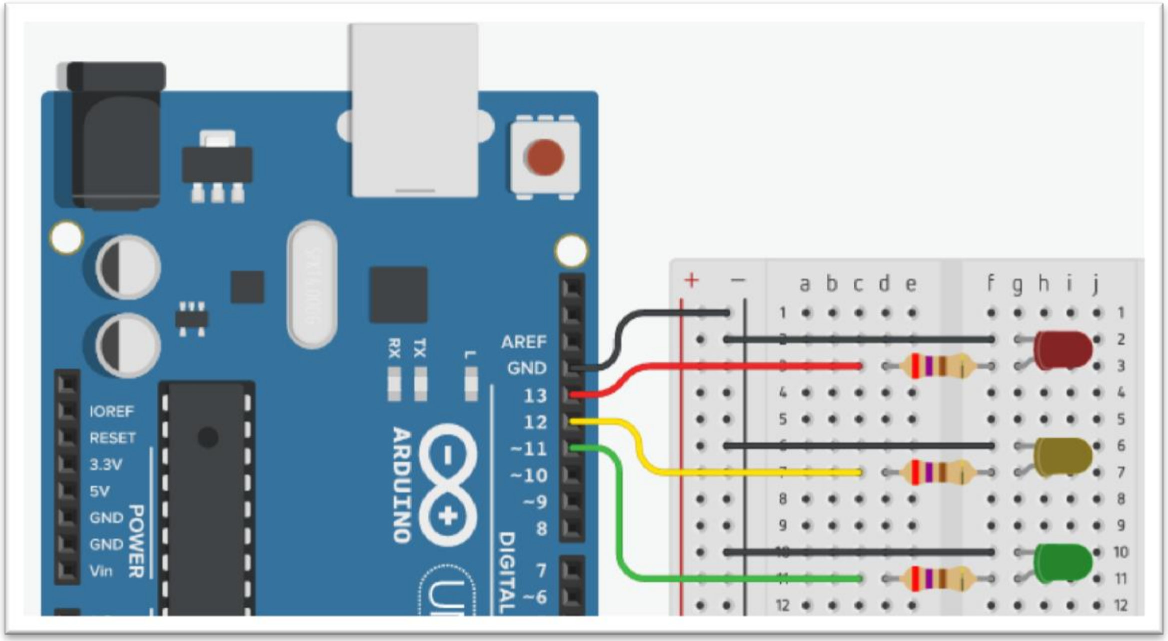
The class can have many **properties** which describe it, and do things by using **methods.**

Level 9 and up

Object-oriented programming

Consider an Arduino program to control a traffic lights

We have 3 LEDs, that can be on or off, and have different colours



Level 9 and up

Object-oriented programming

In object-oriented programming we only need to create **one light class**, a template for how all lights will work. We create light objects, name them and pass a variable.

Instances (objects)

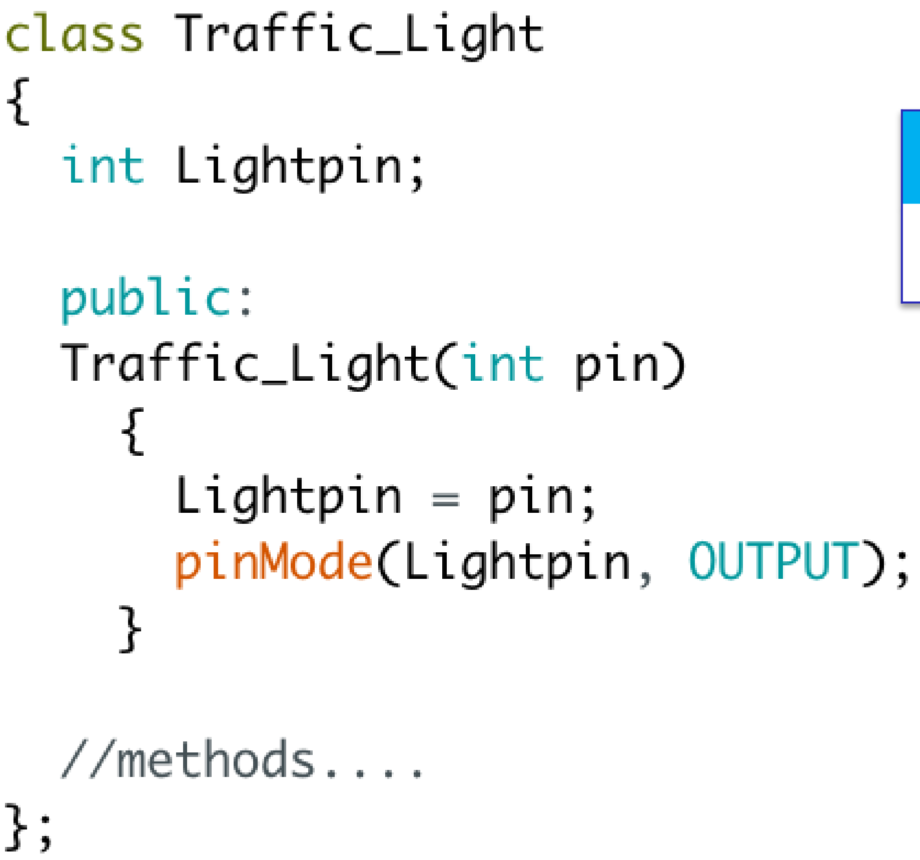
Instance	Property
Red	13
Amber	12
Green	11

Class: Traffic_Light

Properties
Board Pin
Methods
ON (turns light on) OFF (turns light off) Blink (Flashes light on/off)

Level 9 and up

Object-oriented programming



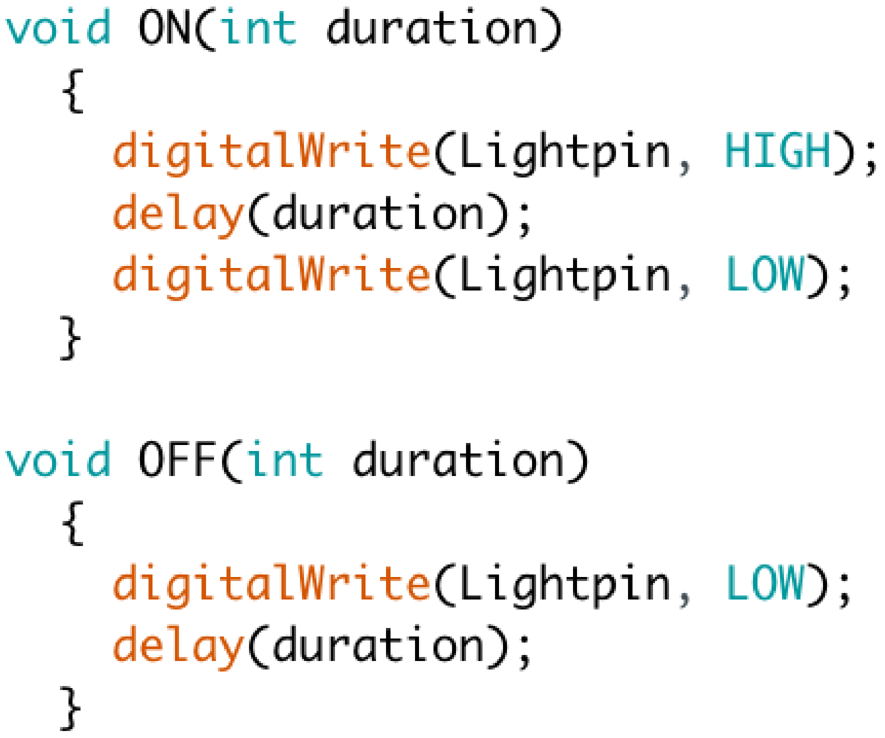
```
class Traffic_Light
{
  int Lightpin;

  public:
  Traffic_Light(int pin)
    {
      Lightpin = pin;
      pinMode(Lightpin, OUTPUT);
    }

  //methods....
};
```

Class: Traffic_Light

Properties
Board Pin

Level 9 and up

Object-oriented programming



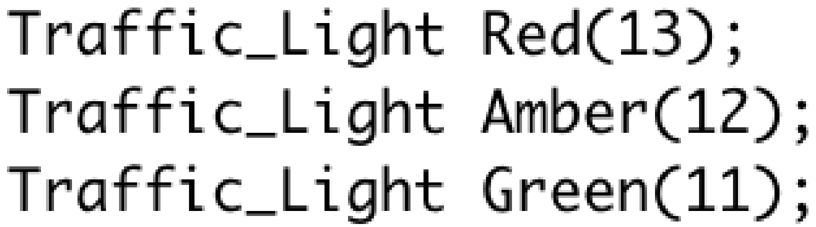
```
void ON(int duration)
  {
    digitalWrite(Lightpin, HIGH);
    delay(duration);
    digitalWrite(Lightpin, LOW);
  }

void OFF(int duration)
  {
    digitalWrite(Lightpin, LOW);
    delay(duration);
  }
```

Class: Traffic_Light

Methods
ON (turns light on) OFF (turns light off)

Level 9 and up

Object-oriented programming

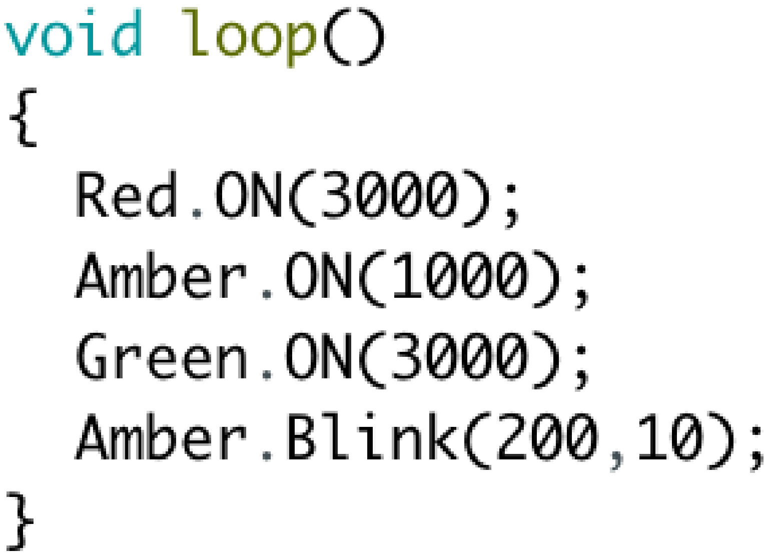


```
Traffic_Light Red(13);
Traffic_Light Amber(12);
Traffic_Light Green(11);
```

Instances (objects)

Name	Property
Red	13
Amber	12
Green	11

Object-oriented programming



```
void loop()
{
   Red.ON(3000);
   Amber.ON(1000);
   Green.ON(3000);
   Amber.Blink(200,10);
}
```

Main Program

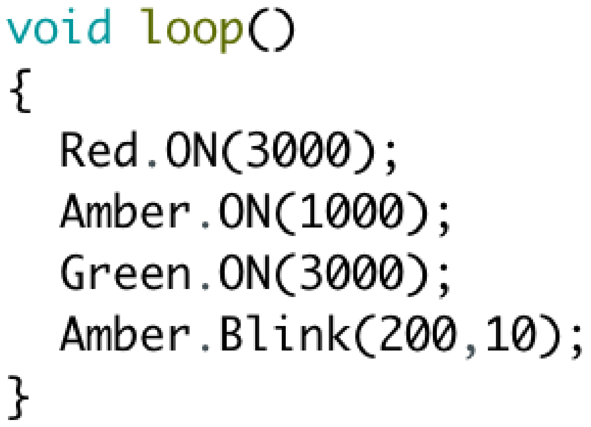
The "run program" is now very clean, short and simple.

Red light on 3 seconds
Amber light on 1 second
Green light on 3 seconds
Amber blinks 10 times

Level 9 and up

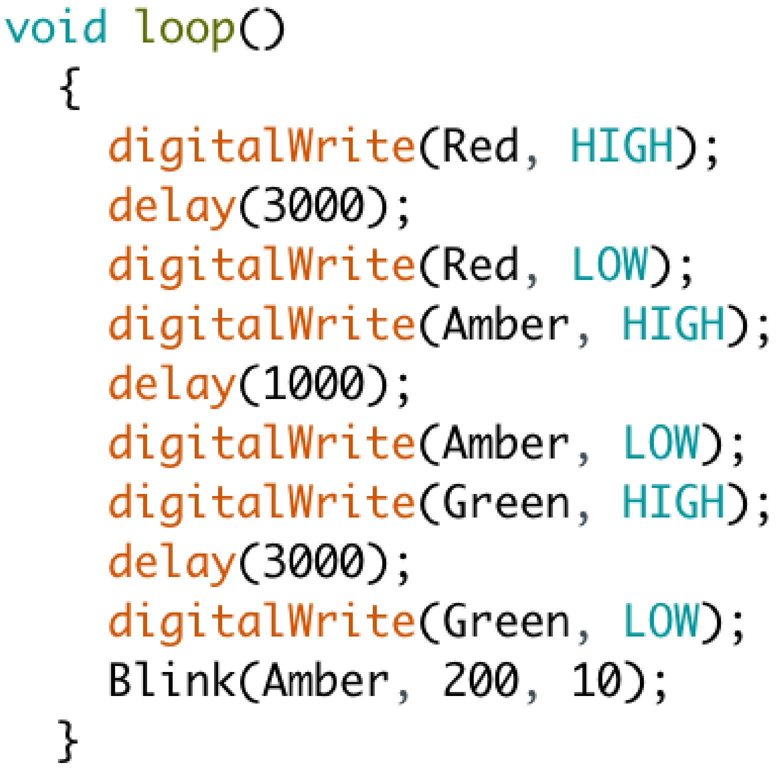
Object-oriented programming

Object-oriented programming



```
void loop()
{
  Red.ON(3000);
  Amber.ON(1000);
  Green.ON(3000);
  Amber.Blink(200,10);
}
```

General-purpose programming



```
void loop()
  {
    digitalWrite(Red, HIGH);
    delay(3000);
    digitalWrite(Red, LOW);
    digitalWrite(Amber, HIGH);
    delay(1000);
    digitalWrite(Amber, LOW);
    digitalWrite(Green, HIGH);
    delay(3000);
    digitalWrite(Green, LOW);
    Blink(Amber, 200, 10);
  }
```

A look at some common programming environments

(visual, general-purpose and object-oriented)

Learning environments

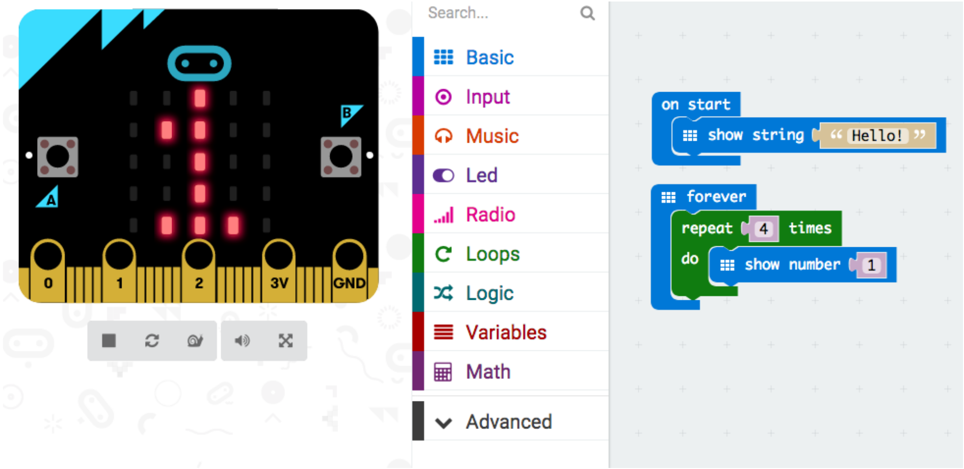
Level	Environment
F-2	Unplugged, BeeBot
3-4	Hopscotch, Hour of Code, Scratch, Sphero
5-6	Scratch, Robots, Code.org, BBC Microbit
7-8	Grok, Khan Academy, CodeHS, BBC Microbit
9-10	Arduino, JavaScript Apps, Python IDLE, Visual Basic

Level 5 and up

BBC Microbit

Visual / JavaScript environment

- https://makecode.microbit.org/



```
on start
    show string "Hello!"

forever
    repeat 4 times
    do
        show number 1
```

Python environment

- https://python.microbit.org

Learning environments

Advantages: Comprehensive environment, videos, tests, examples, student development area

Disadvantages: Often too much content, not focused on our curriculum, students can get lost

Environments advice

Use for programming practice, acquiring skills, homework tasks and threshold activities

Direct the students to the most relevant sections

Do the modules yourself before the students

Set your own programming assessment tasks

Level 7 and up

Suitable languages

Needs to be a general-purpose programming language from Level 7 up

But students can use visual programming if they are low (concepts are more important than syntax)

Popular languages are:

- Python
- JavaScript
- Arduino C++

Considerations

What is your goal? - this should drive everything

Bottom line, try different languages and go with the one you feel most comfortable with

Consider getting someone to mentor you

Considerations

Regardless of the programming language the fundamental thinking is crucial

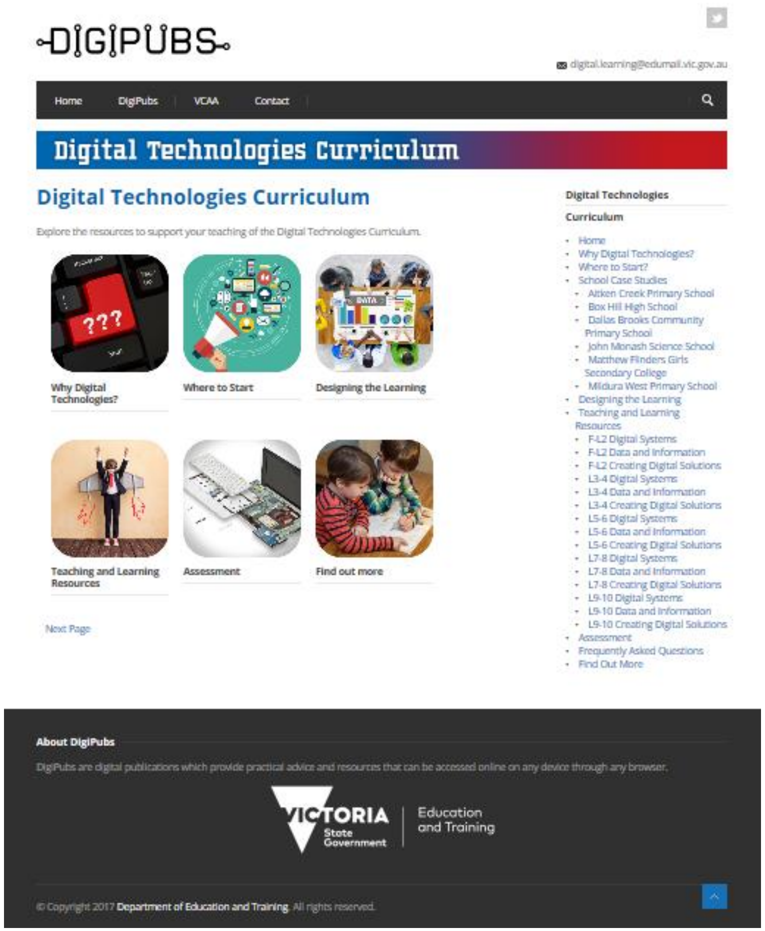
If students grasp the concepts transferring to a different language is fairly easy, they only need to learn syntax (spelling and grammar)

Consider using posters of loops and if statements in different programming languages

Resources

http://www.digipubs.vic.edu.au/curriculum/digitaltechnologies/digital-technologies-curriculum

DigiPubs



DIGIPUBS

digital.learning@edumail.vic.gov.au

Home | DigiPubs | VCAA | Contact

Digital Technologies Curriculum

Digital Technologies Curriculum

Explore the resources to support your teaching of the Digital Technologies Curriculum.

Why Digital Technologies?

Where to Start

Designing the Learning

Teaching and Learning Resources

Assessment

Find out more

Next Page

Digital Technologies Curriculum

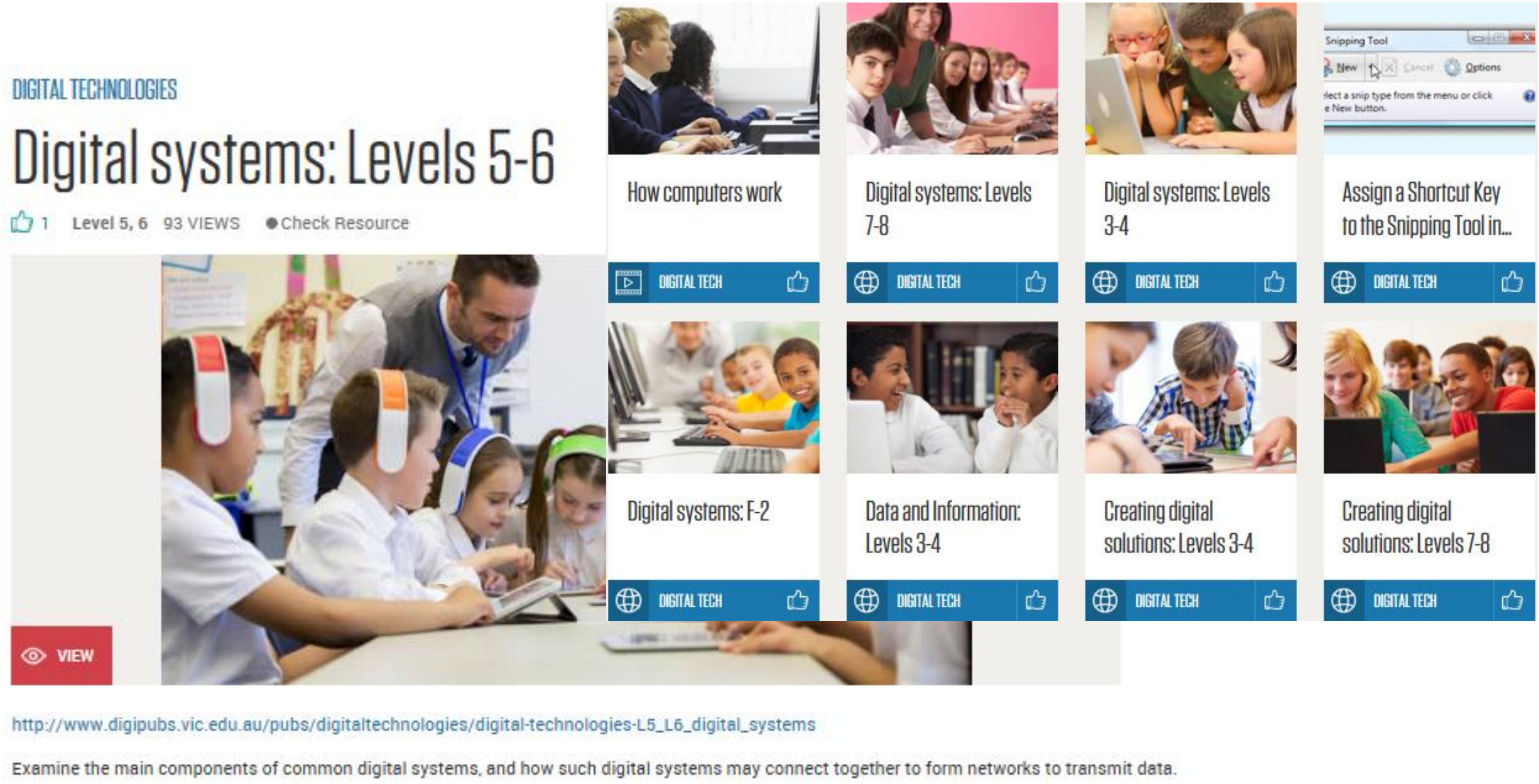
- Home
- Why Digital Technologies?
- Where to Start?
- School Case Studies
  - Aitken Creek Primary School
  - Box Hill High School
  - Dallas Brooks Community Primary School
  - John Monash Science School
  - Matthew Flinders Girls Secondary College
  - Mildura West Primary School
- Designing the Learning
- Teaching and Learning Resources
  - F-L2 Digital Systems
  - F-L2 Data and Information
  - F-L2 Creating Digital Solutions
  - L3-4 Digital Systems
  - L3-4 Data and Information
  - L3-4 Creating Digital Solutions
  - L5-6 Digital Systems
  - L5-6 Data and Information
  - L5-6 Creating Digital Solutions
  - L7-8 Digital Systems
  - L7-8 Data and Information
  - L7-8 Creating Digital Solutions
  - L9-10 Digital Systems
  - L9-10 Data and Information
  - L9-10 Creating Digital Solutions
- Assessment
- Frequently Asked Questions
- Find Out More

About DigiPubs

DigiPubs are digital publications which provide practical advice and resources that can be accessed online on any device through any browser.

VICTORIA State Government | Education and Training https://fuse.education.vic.gov.au/

Fuse



DIGITAL TECHNOLOGIES

Digital systems: Levels 5-6

1 Level 5, 6 93 VIEWS ● Check Resource

VIEW

http://www.digipubs.vic.edu.au/pubs/digitaltechnologies/digital-technologies-L5_L6_digital_systems

Examine the main components of common digital systems, and how such digital systems may connect together to form networks to transmit data.

How computers work — DIGITAL TECH

Digital systems: Levels 7-8 — DIGITAL TECH

Digital systems: Levels 3-4 — DIGITAL TECH

Assign a Shortcut Key to the Snipping Tool in... — DIGITAL TECH

Digital systems: F-2 — DIGITAL TECH

Data and Information: Levels 3-4 — DIGITAL TECH

Creating digital solutions: Levels 3-4 — DIGITAL TECH

Creating digital solutions: Levels 7-8 — DIGITAL TECH

VCAA Professional Learning Support

To find online webinars or face-to-face sessions in your area:

http://www.vcaa.vic.edu.au/Pages/foundation10/viccurriculum/viccur-proflearn-specialists.aspx

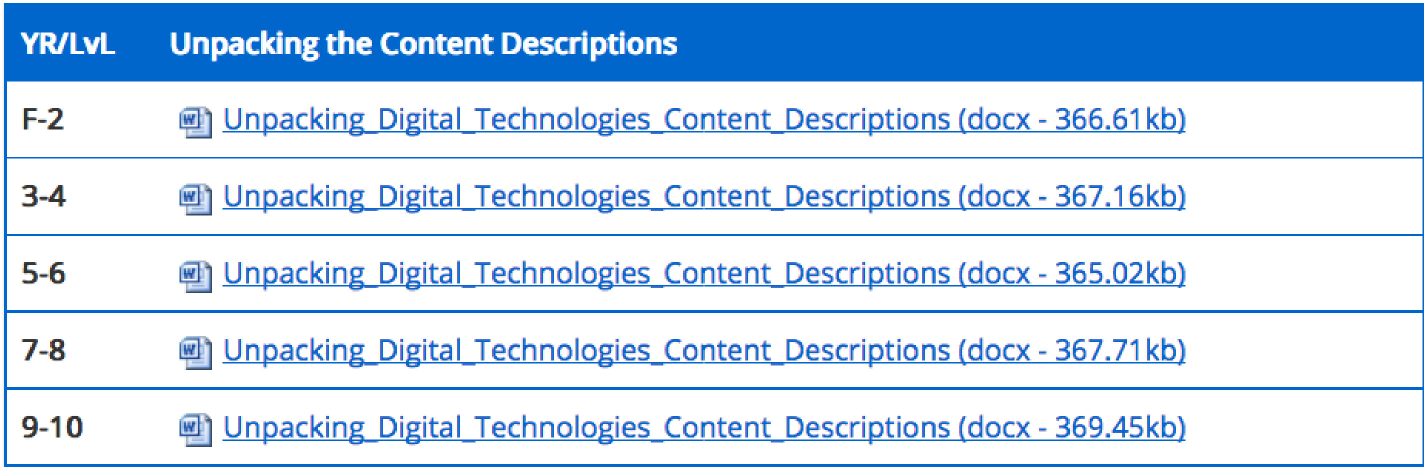
To request a session for your local network:

http://www.vcaa.vic.edu.au/Pages/foundation10/viccurriculum/viccur-proflearn-specialists.aspx#request

Teaching Resources

The VCAA have some model lesson activities and sequences online for teachers to use

http://www.vcaa.vic.edu.au/Pages/foundation10/viccurriculum/digitech/teachresources.aspx



YR/LvL	Unpacking the Content Descriptions
F-2	Unpacking_Digital_Technologies_Content_Descriptions (docx - 366.61kb)
3-4	Unpacking_Digital_Technologies_Content_Descriptions (docx - 367.16kb)
5-6	Unpacking_Digital_Technologies_Content_Descriptions (docx - 365.02kb)
7-8	Unpacking_Digital_Technologies_Content_Descriptions (docx - 367.71kb)
9-10	Unpacking_Digital_Technologies_Content_Descriptions (docx - 369.45kb)

Questions / Feedback

Has this presentation developed your understanding of the Digital Technology curriculum?

How can we help?

Other comments?

Your feedback is important to us

It will help us plan future sessions.

Please take some time to complete an evaluation of this session.

https://vcaa.qualtrics.com/jfe/form/SV_bdsrTF9JFeL38Q5

Daryl Croke
VCAA Specialist Teacher (Digital Technologies)
Teacher – Mount Ridley P-12 College, Craigieburn