Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Leaving Certificate Computer Science
National Workshop 2

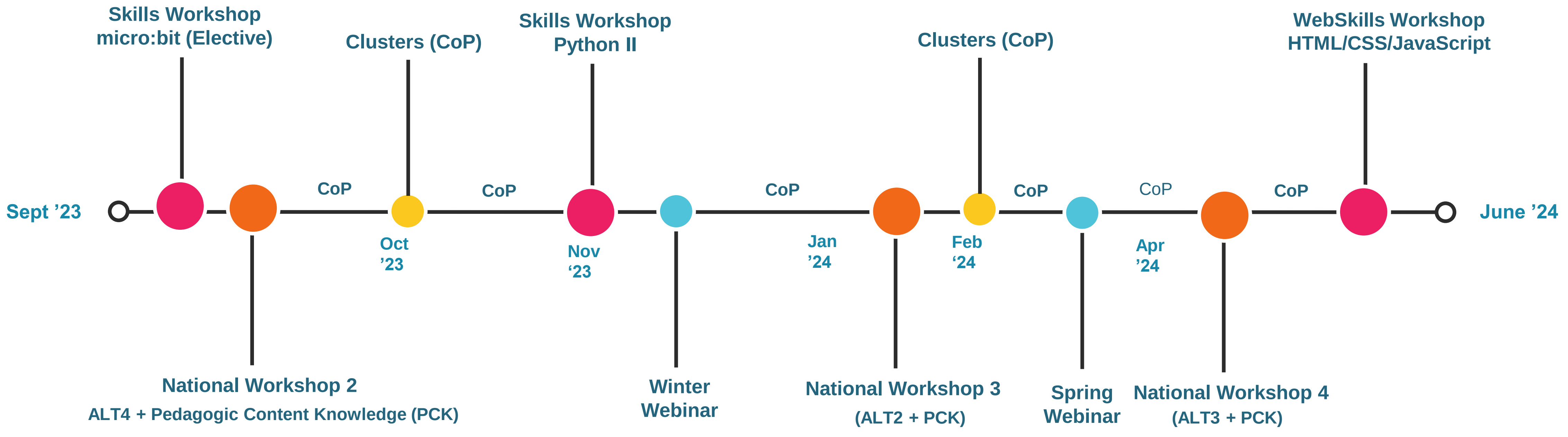
Day 1

Workshop Overview

Session 1 10:00 - 11:30	Introduction Computer Systems I
Tea/Coffee 11:30 – 12:00	
Session 2 12:00 - 13:30	Computational Thinking II
Lunch 13:30 - 14:30	
Session 3 14:30 - 16:30	PRIMM and Curriculum Planning

Dates for your Diary for 2023/4

Oide



Next CPD event: Community of Practice cluster meetings – online early November

Oide

Introducing Oide

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

Lárionad Ceannaireachta Scoile

CSL

Centre for School Leadership

An tSraith Shóisearach do Mhúinteoirí

Junior CYCLE for teachers

NIPT

An Clár Náisiúnta Ionduchtaithe do Mhúinteoirí

The National Induction Programme for Teachers

PDST

Professional Development Service for Teachers | An tSeirbhís um Fhorbairt Ghairmiúil do Mhúinteoirí

Supports Provided by Oide

National Workshops	Webinars	School Support	Scoilnet
Skills Workshops	Collaboratives	Oide website	CompSci

Purpose for the Day

To allow Phase 5 LCCS teachers to engage with the core concepts of Computer Systems and Computational Thinking.

To experience ALT4 (Embedded Systems) through the eyes of the student by engaging with the Design Process.

Key Messages

All learning outcomes (LOs) are interwoven. This means that the specification can be used in many different ways.

ALTs provide an opportunity to teach theoretical aspects of LCCS.

LCCS can be mediated through a constructivist pedagogical approach.

Group work is a key feature in the teaching, learning and assessment of LCCS.

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

LCCS NW2 Session 1

Number Systems

By the end of this session..

Participants will be enabled to…

- develop an understanding of Computational Thinking concepts such as abstraction, decomposition, algorithmic thinking and pattern recognition
- develop a shared understanding of how *programming as a process* can be used to mediate CT in the classroom
- convert decimal numbers to binary numbers and vice versa

Computational Thinking

"… the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent."
(Wing 2011)

Decomposition of a decimal number

	powers of 10	4	3	2	1	0	
Most Significant Digit →		7	0	2	8	4	← Least Significant Digit

Digit	Place	Value
4	Units	$4 \times 10^0 = 4$
8	Tens	$8 \times 10^1 = 80$
2	Hundreds	$2 \times 10^2 = 200$
0	Thousands	$0 \times 10^3 = 0$
7	Ten Thousands	$7 \times 10^4 = 70000$

etc.

70,000 + 0,000 + 200 + 80 + 4 = 70,284

What about a binary number?

P4

Decomposition of a binary number

Most Significant Bit (MSB)

4 3 2 1 0 ← powers of 2

1 0 0 1 1 ← Least Significant Bit (LSB)

Units	$1 \times 2^0 = 1$
Twos	$1 \times 2^1 = 2$
Fours	$0 \times 2^2 = 0$
Eights	$0 \times 2^3 = 0$
Sixteens	$1 \times 2^4 = 16$

Can we do this in Python?

16 + 0 + 0 + 2 + 1 = 19

etc.

$$10011_2 = 19_{10}$$

P4

Convert 19_{10} to base 2

$$\frac{\text{Dividend}}{\text{Divisor}} = \text{Quotient} + \text{Remainder}$$

Oide

1. Divide by 2 …. note the remainder
2. The quotient becomes the new dividend
3. Keep dividing …
4. Stop when the quotient reaches zero
5. Read the answer from the bottom up

$$\frac{19}{2} = 9 + 1$$

$$\frac{9}{2} = 4 + 1$$

$$\frac{4}{2} = 2 + 0$$

$$\frac{2}{2} = 1 + 0$$

$$\frac{1}{2} = 0 + 1$$

So, $19_{10} = 10011_2$

2	19	
2	9	+ 1
2	4	+ 1
2	2	+ 0
2	1	+ 0
2	0	+ 1

Decimal -> Binary (another example)

Oide

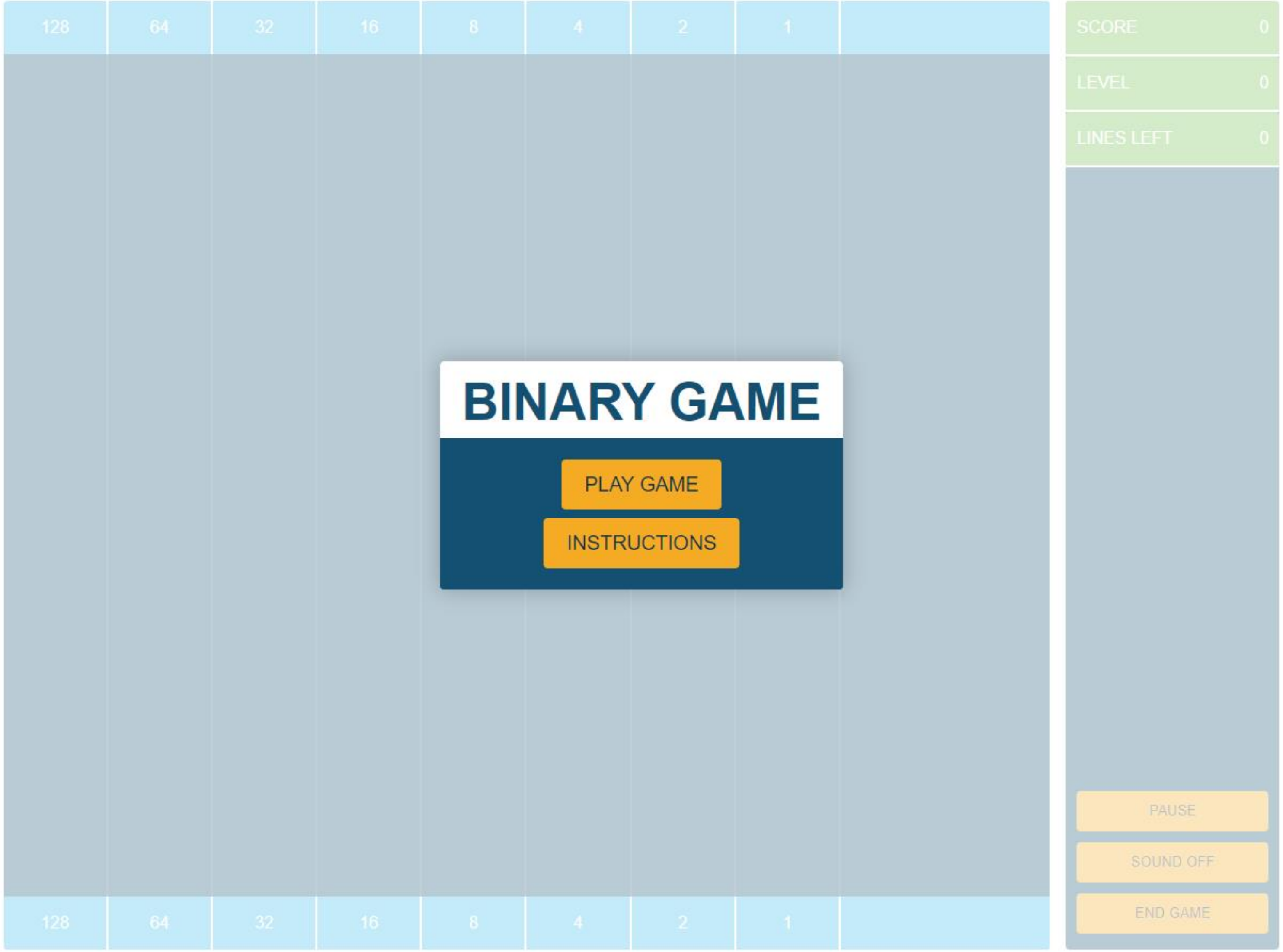
1. Divide by 2 …. note the remainder
2. The quotient becomes the new dividend
3. Keep dividing …
4. Stop when the quotient reaches zero
5. Read the answer from the bottom up

Convert 47_{10} to base 2

2	47
2	23 + 1
2	11 + 1
2	5 + 1
2	2 + 1
2	1 + 0
2	0 + 1

$47_{10} = 101111_2$

P5

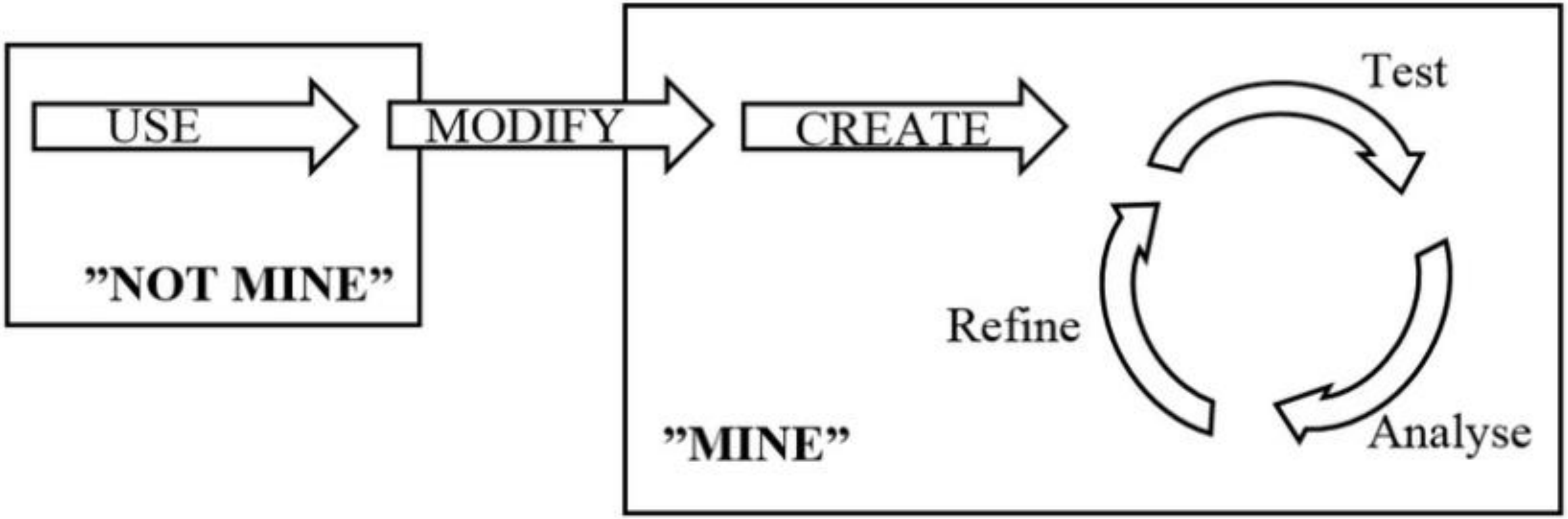


https://learningcontent.cisco.com/games/binary/index.html

Code Along Activity

Oide

Use Modify Create



Program Tracing / Debugging

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

```python
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

Program Tracing / Debugging

Oide

The Notional Machine / Working Memory

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

This is what is displayed ….

Program Tracing / Debugging


```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: 9

This is what is displayed ….

Program Tracing / Debugging

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

```
quotient:    9
remainder1:  1
```

This is what is displayed ….

Program Tracing / Debugging

Oide

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)
# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2
# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

```
quotient:   9
remainder1: 1
```

This is what is displayed ….

```
>>> 9 1
```

Program Tracing / Debugging

Oide

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: ~~9~~ 4

remainder1: 1

This is what is displayed ….

>>> 9 1

Program Tracing / Debugging

Oide

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: ~~9~~ 4

remainder1: 1

remainder2: 1

This is what is displayed ….

>>> 9 1

Program Tracing / Debugging

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: ~~9~~ 4

remainder1: 1

remainder2: 1

This is what is displayed ….

>>> 9 1
>>> 4 1

Program Tracing / Debugging

Oide

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: ~~9~~ ~~4~~ 2

remainder1: 1

remainder2: 1

This is what is displayed ….

```
>>> 9 1
>>> 4 1
```

Program Tracing / Debugging

Oide

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: ~~9~~ ~~4~~ 2

remainder1: 1

remainder2: 1

remainder3: 0

This is what is displayed ….

```
>>> 9 1
>>> 4 1
```

Program Tracing / Debugging

```
quotient = 19//2
remainder1 = 19%2
print(quotient, remainder1)

# copy+paste ...
quotient = 9//2
remainder2 = 9%2
print(quotient, remainder2)
# Once ...
quotient = 4//2
remainder3 = 4%2
print(quotient, remainder3)
# Twice ...
quotient = 2//2
remainder4 = 2%2

# Three times ...
quotient = 1//2
remainder5 = 1%2
```

The Notional Machine / Working Memory

quotient: ~~9~~ ~~4~~ 2

remainder1: 1

remainder2: 1

remainder3: 0

This is what is displayed ….

```
>>> 9 1
>>> 4 1
>>> 2 0
```

Group Activity: Breakout

Binary -> Decimal (1 of 2)

```
binary_number = 10011
decimal_number = 0

digit0 = 10011 % 10 # lsb
stem = 10011 // 10
print(stem, digit0)
```

How could we develop this Python code to a general solution?

Binary -> Decimal

```
# ... convert binary 10011 to decimal ...
# ... the initial number is a string
binary_number = "10011"
# index:          01234

units    = int(binary_number[4])*1
twos     = int(binary_number[3])*2
fours    = int(binary_number[2])*4
eights   = int(binary_number[1])*8
sixteens = int(binary_number[0])*16
decimal = units+twos+fours+eights+sixteens
```

How could we develop this Python code to a general solution?

20 minute breakout

Break

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

LCCS NW2 Session 2

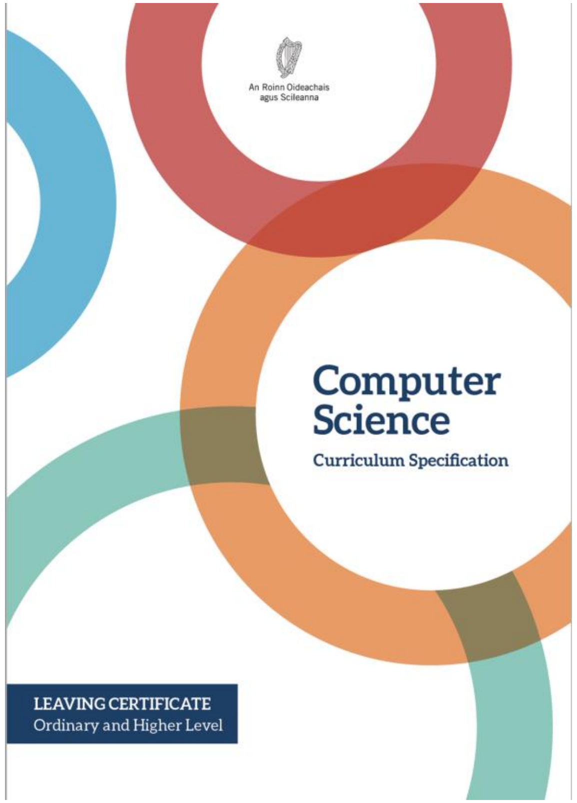
Computational Thinking II

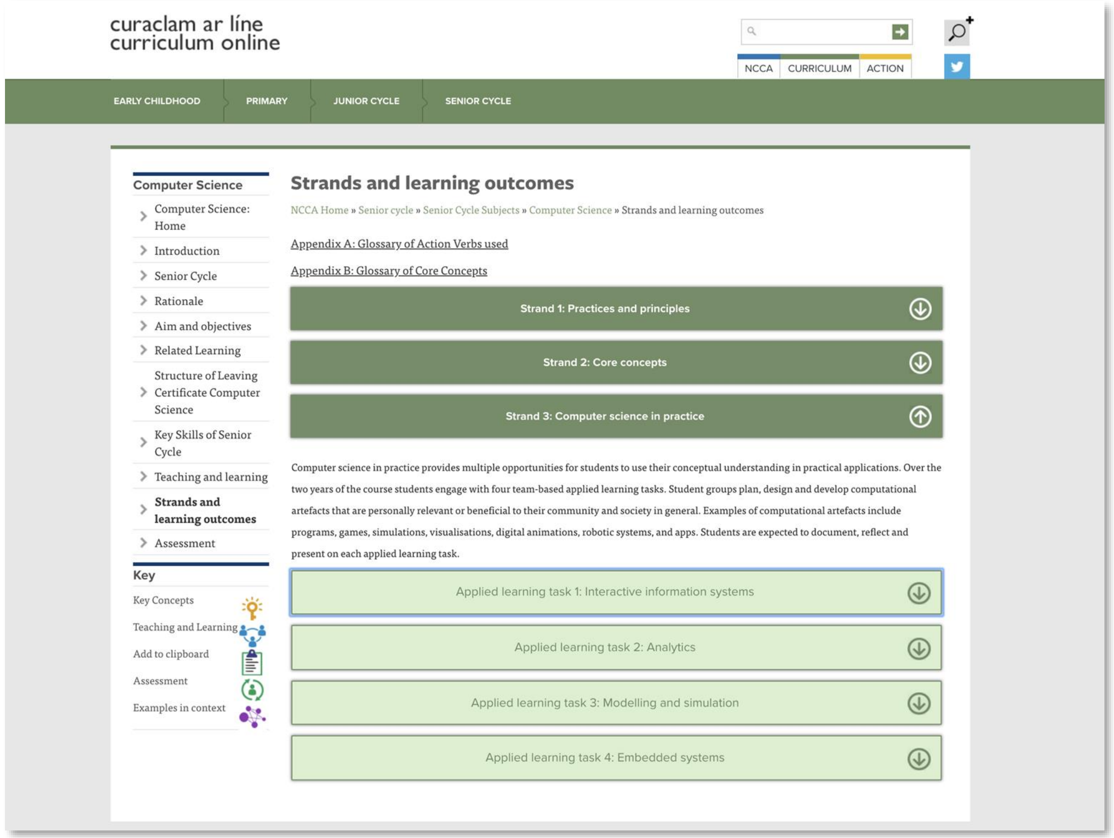
By the end of this session …

Participants will have been enabled to…

- develop their understanding of Computational Thinking (CT) concepts
- consider the questions: What is CT? Why is CT important?
- reflect on successful pedagogies for teaching CT skills
- analyse and develop solutions to problems of various types using CT skills such as abstraction, decomposition, pattern recognition and algorithmic thinking

LCCS Curriculum Specification





https://www.curriculumonline.ie

What does the specification say?

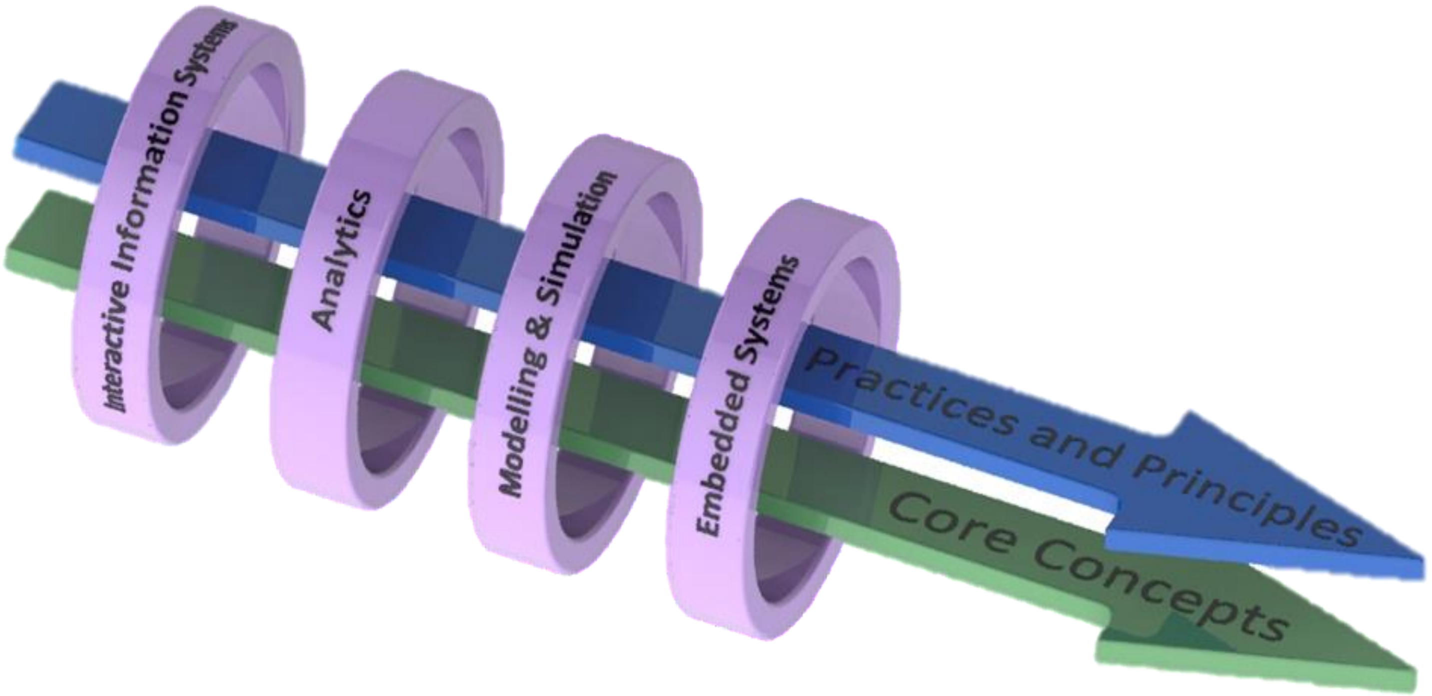
*"Computer science is the study of computers and algorithmic processes. Leaving Certificate Computer Science includes how programming and **computational thinking** can be applied to the solution of problems, and how computing technology impacts the world around us."*

[LCCS Spec. Page 2, paragraph 1]

Strand 1: Practices and principles	Strand 2: Core concepts	Strand 3: Computer science in practice
‣ Computers and society ‣ Computational thinking ‣ Design and development	‣ Abstraction ‣ Algorithms ‣ Computer systems ‣ Data ‣ Evaluation/Testing	‣ Applied learning task 1 - Interactive information systems ‣ Applied learning task 2 - Analytics ‣ Applied learning task 3 - Modelling and simulation ‣ Applied learning task 4 - Embedded systems

What does the specification say?

"The role of programming in computer science is like that of practical work in the other subjects — it provides motivation, and a context within which ideas are brought to life. Students learn programming by solving problems through ***computational thinking*** *processes and through practical applications such as applied learning tasks."* LCCS specification (2017)



What is Computational Thinking?

"Computational Thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent."

Jeannette M. Wing
Carnegie Mellon University (2011)

Computational Thinking Concepts

Oide

Source: https://csunplugged.org/en/computational-thinking/

Simple Daily Examples

Looking up a name in an alphabetically sorted list

Linear: start at the top

Binary search: start in the middle

Standing in a queue at a bank, supermarket, check in desk, passport control

Performance analysis of task scheduling

Taking your children to football, music and the swimming pool

Traveling salesman (with more constraints)

Cooking a gourmet meal

Multi-tasking, Parallel processing:

Cleaning out your garage

Keeping only what you need vs. throwing out stuff when you run out of space.

Storing away your child's toys scattered on the floor

Using hashing (e.g., by shape, by color)

Why is Computational Thinking Important?

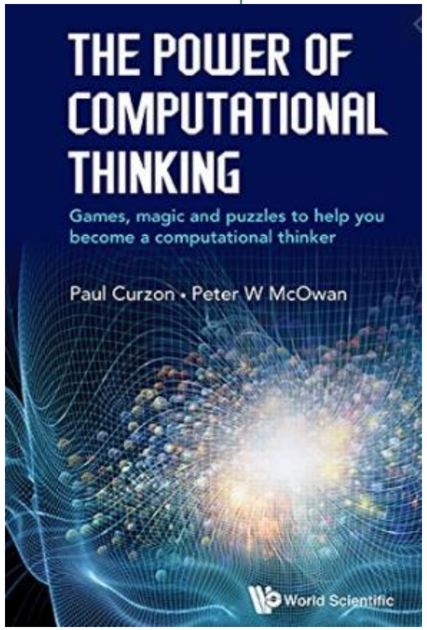
- It moves students beyond being technologically literate
- It creates problem solvers instead of software technicians
- It emphasises the creation of knowledge rather than the use of information
- It presents endless possibilities for creative problem solving
- It enhances the problem-solving techniques you already teach

(Source: Pat Phillips, NECC 2007, Atlanta)

Oide

“What are effective ways for teaching computational thinking?”

How to Teach Computational Thinking



- ❑ Increase your own CT knowledge
- ❑ Integrate CT concepts into everyday instruction
- ❑ Use CT terms for everyday tasks

e.g. “Let’s create an algorithm for …”

- ❑ Encourage students to formulate and test their own hypotheses

e.g. “Crime rates are on the rise …”

- ❑ Provide opportunities for students to transfer their learning to other situations

Successful CT Pedagogies

- ❑ Analogy / Storytelling
- ❑ CS Unplugged
  - Kinaesthetic
  - Role Playing
  - Puzzles
  - Art
  - Games
  - Magic
- ❑ Enquiry Based Learning (TEMI)

Programming Practice (Python / JavaScript)

Oide

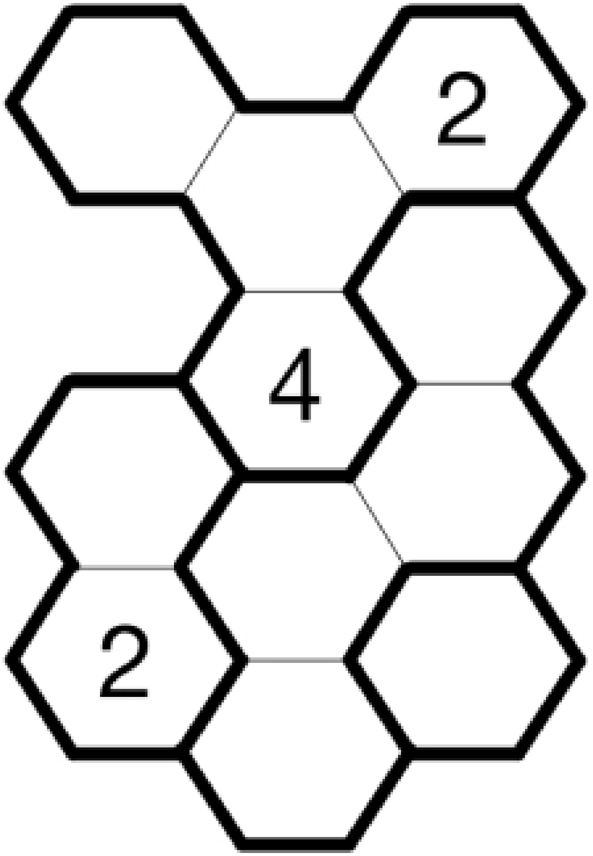
Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

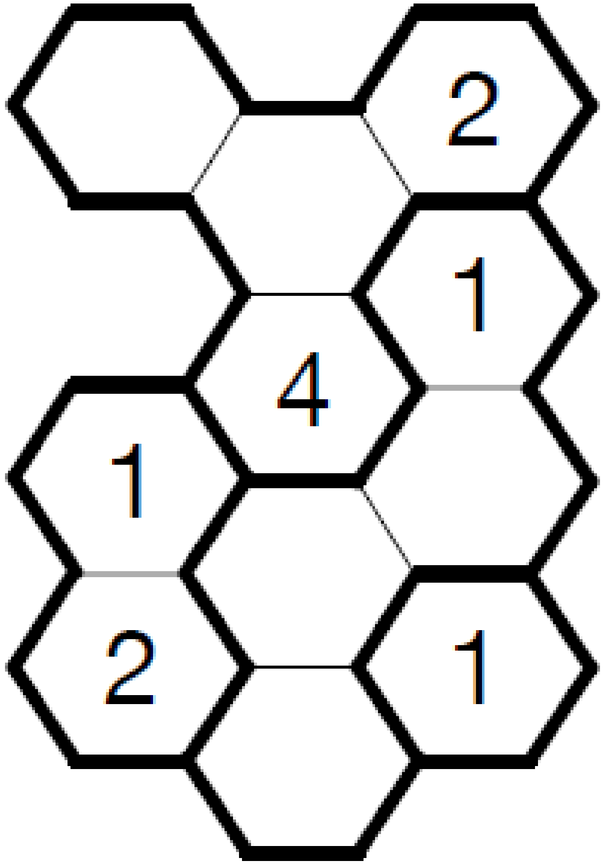
Supporting the Professiona Learning of School Leaders and Teachers

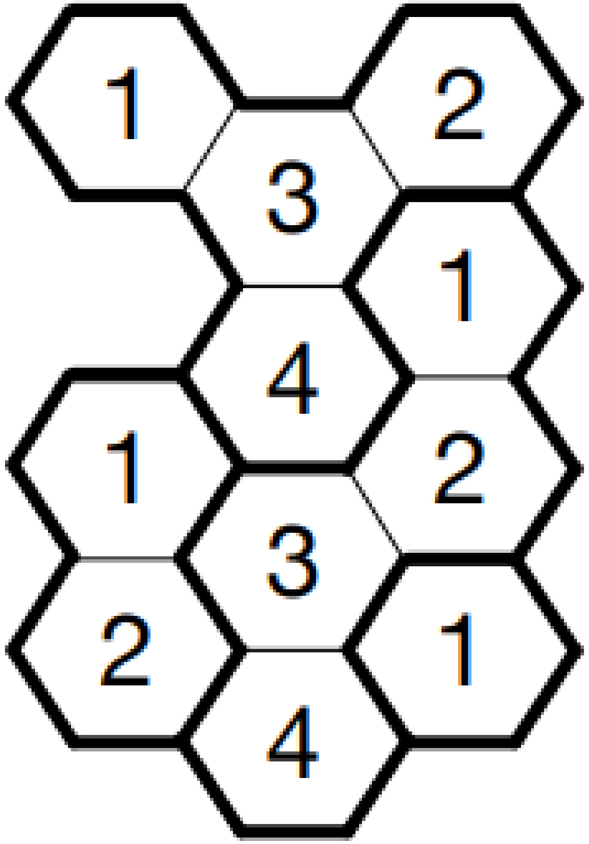
Applying Computational Thinking Skills

Examples

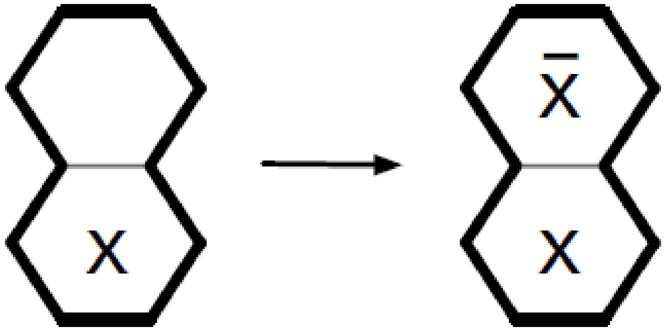
Cut Hive Logic Puzzles



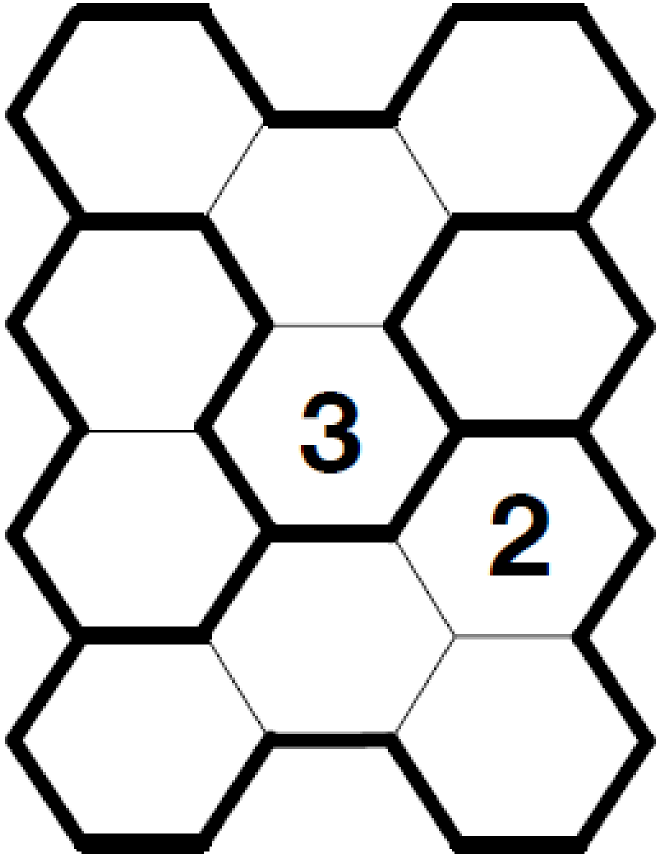


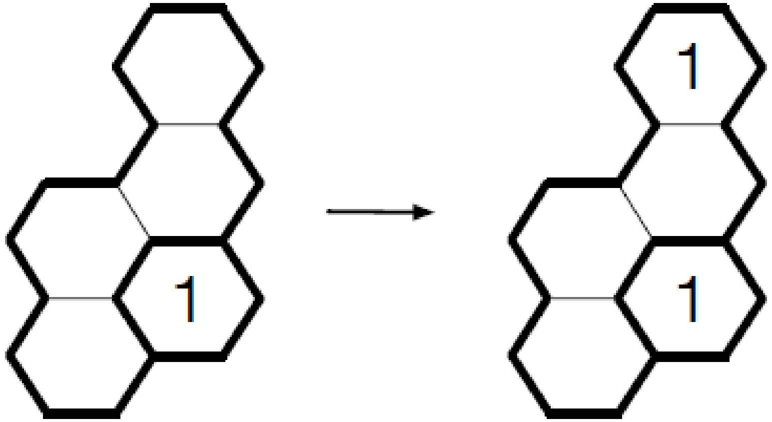


Cut Hive Logic Puzzles



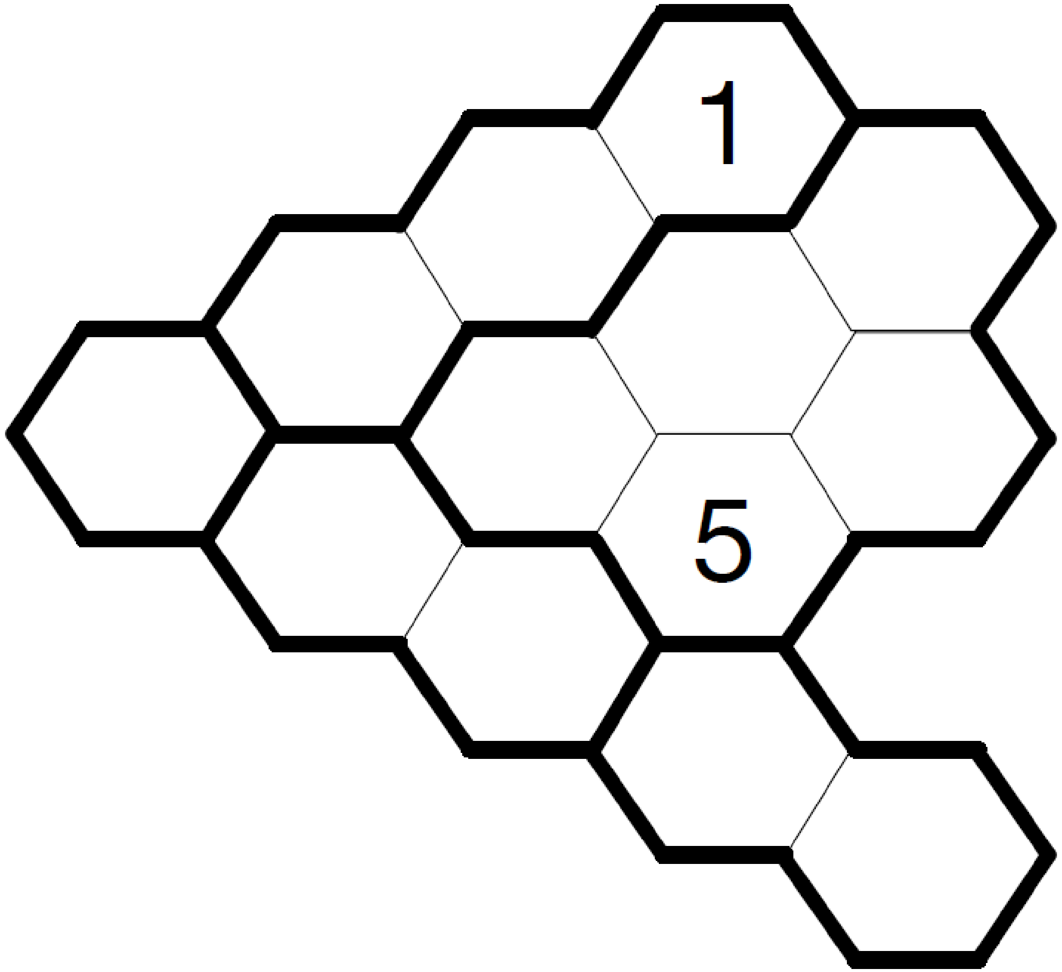
Single Hexagon



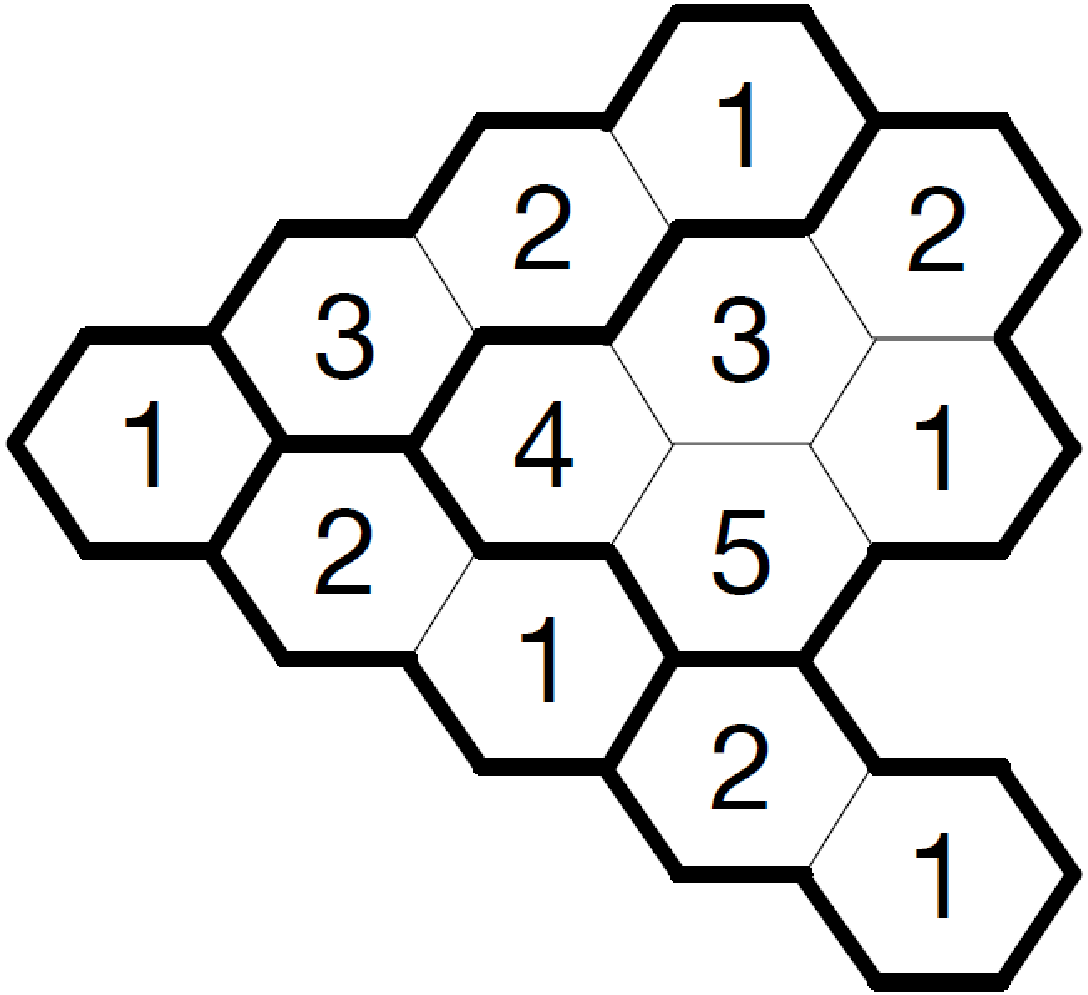


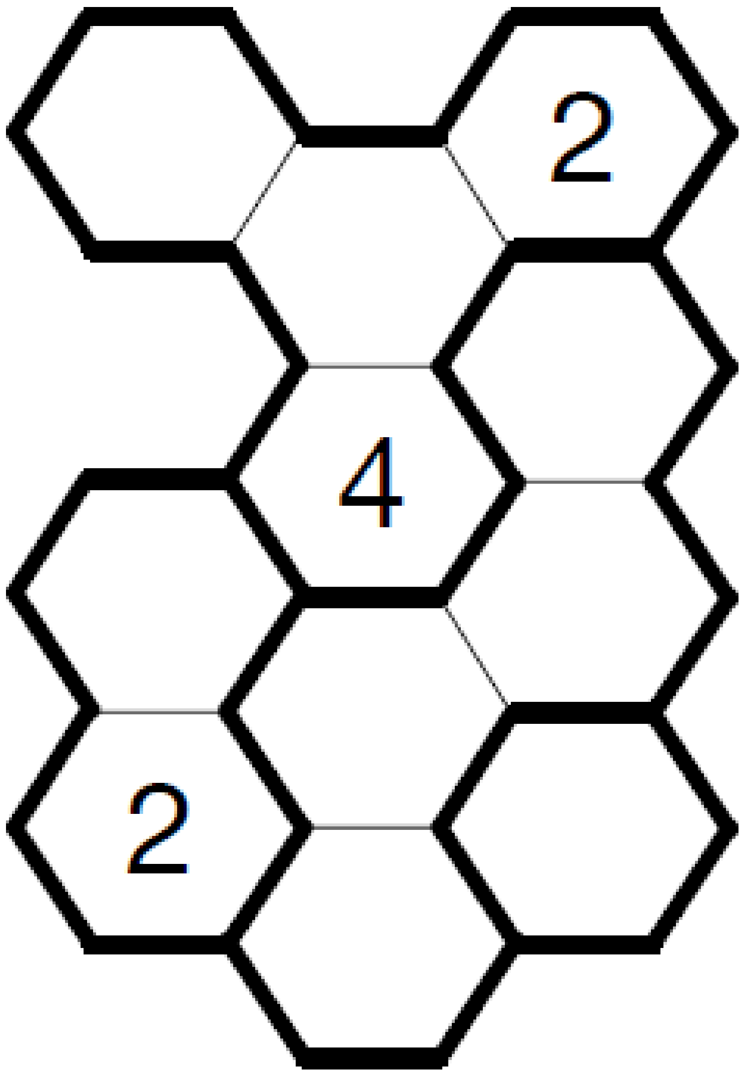
Corners

Challenge



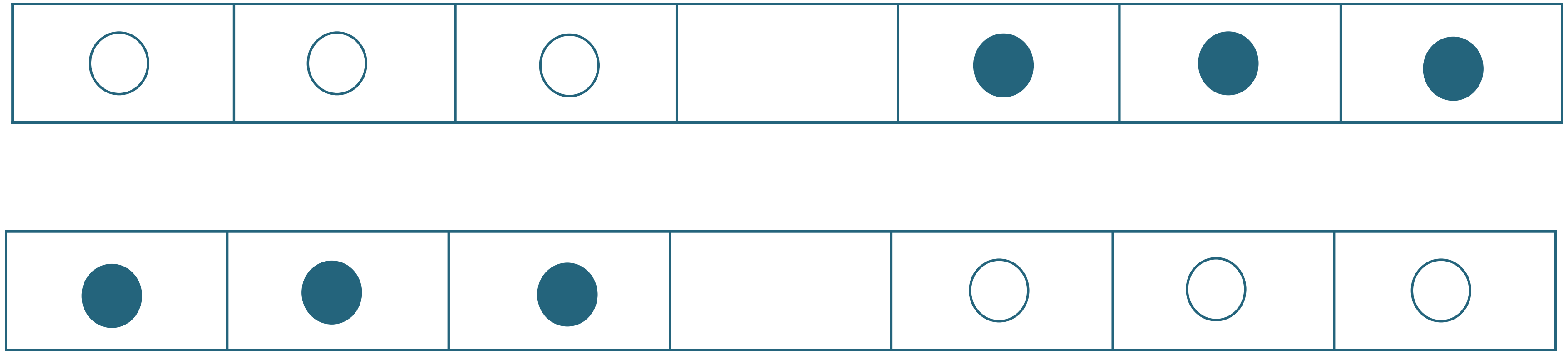
Solution




2
4
2

Algorithmic Thinking

The aim is swap the positions of the black and white pieces.



Pieces can move either by sliding into an adjacent empty square, or by jumping a single adjacent piece into the empty square immediately beyond.

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

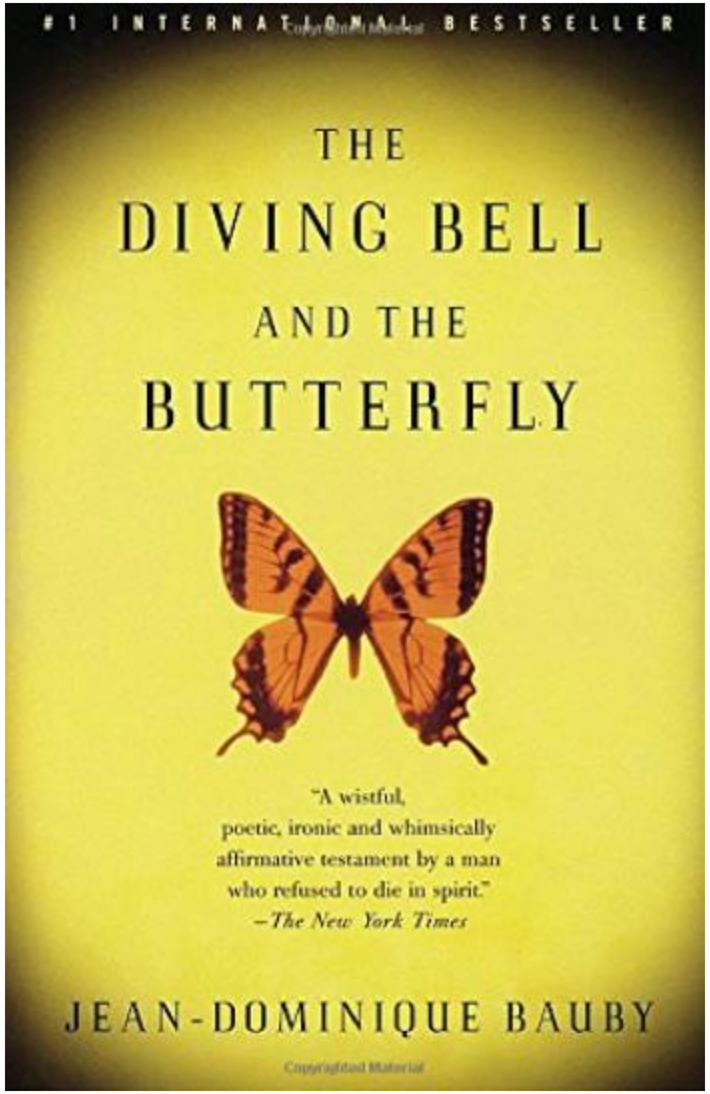
Supporting the Professiona Learning of School Leaders and Teachers

Group Activity

Scenarios

Group Activity

Scenario 1 (Storytelling)

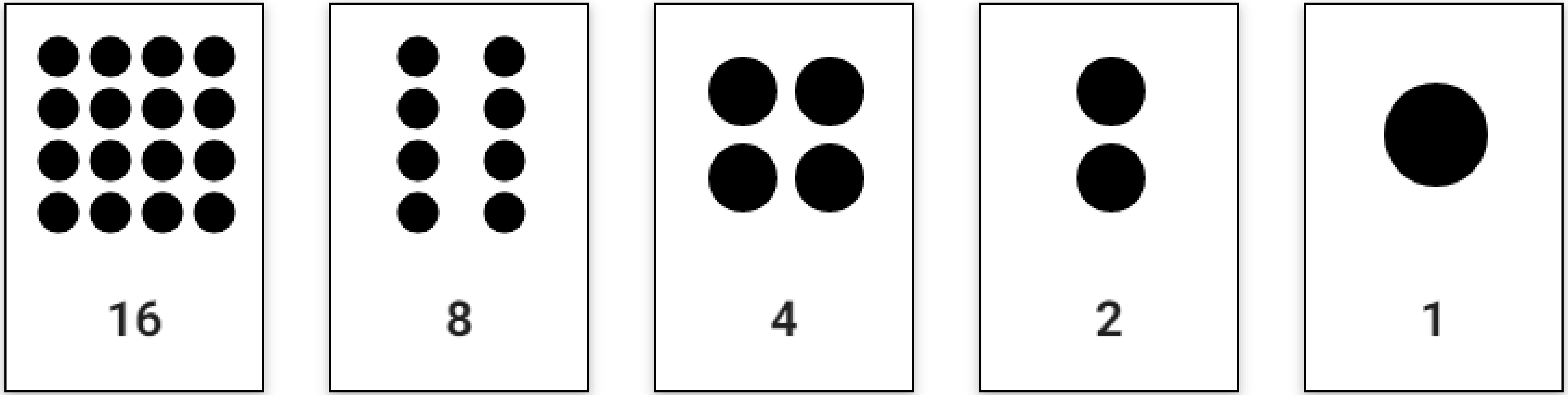


https://www.youtube.com/watch?v=t4Ek4ZBpshs

'The Diving Bell and the Butterfly' is an incredibly uplifting book. It's the autobiography of Jean-Dominique Bauby, written after he woke up in a hospital bed totally paralysed. In the book, he describes life with locked-in syndrome. He did have a way to communicate not only to write the book but also with medics, friends and family. He did it without any technology at all. How?

Scenario 2 (Kinaesthetic)

Oide

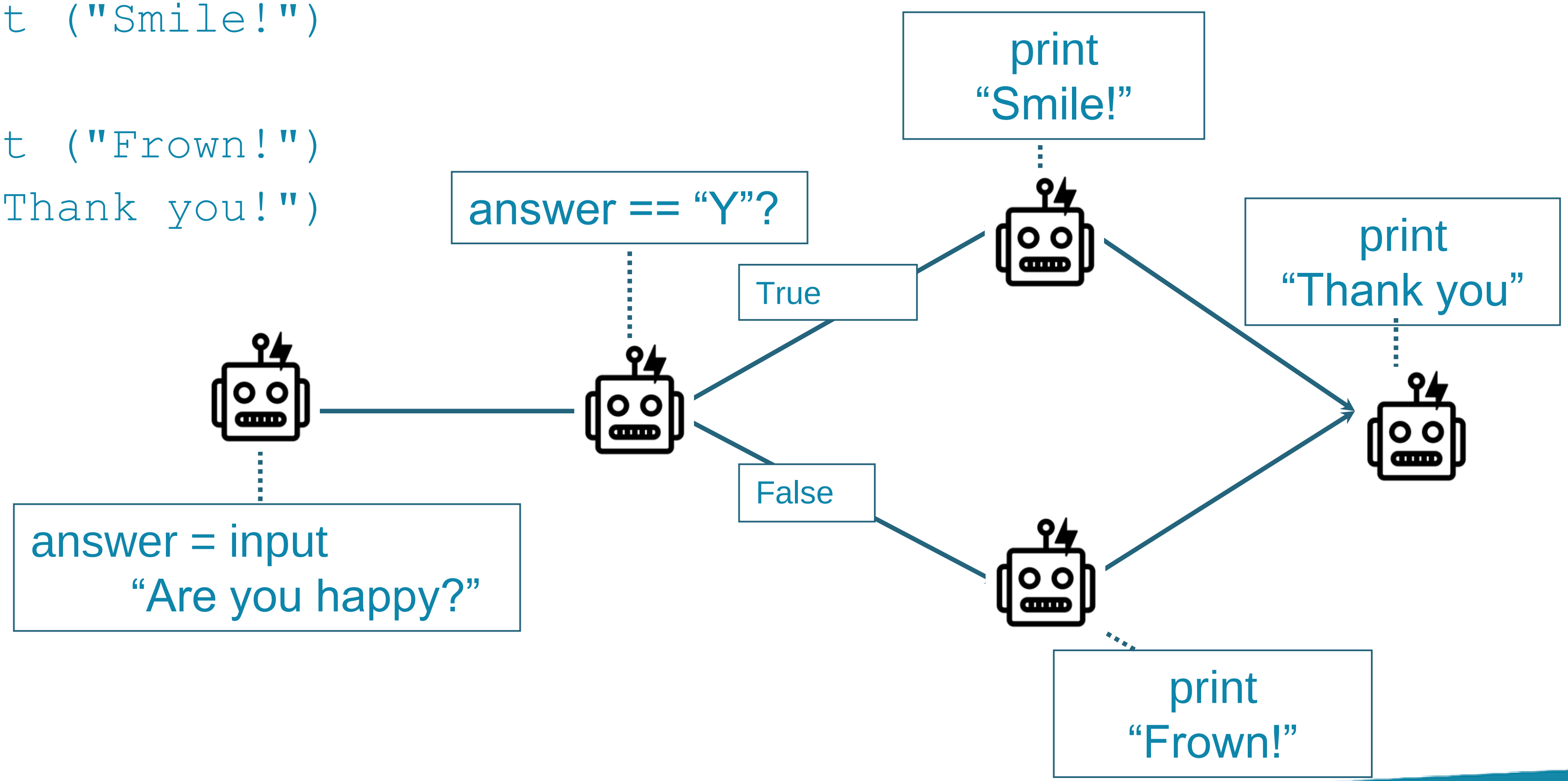


Which cards do we need to turn over to make the number 13?

(The cards are blank on the reverse side.)

Scenario 3 (Role play)

```
answer = input ("Are you happy?")
if answer == "Y":
    print ("Smile!")
else:
    print ("Frown!")
print ("Thank you!")
```



Instructions

In your assigned group go to the breakout area

Read the scenario provided

Design a presentation based on the scenario …

- a description of the scenario provided
- a demonstration of the activity
- an outline of how the pedagogy could be used to teach CT concepts
- suggestions on how the scenario could be used (or extended) to design lesson(s) suitable for LCCS

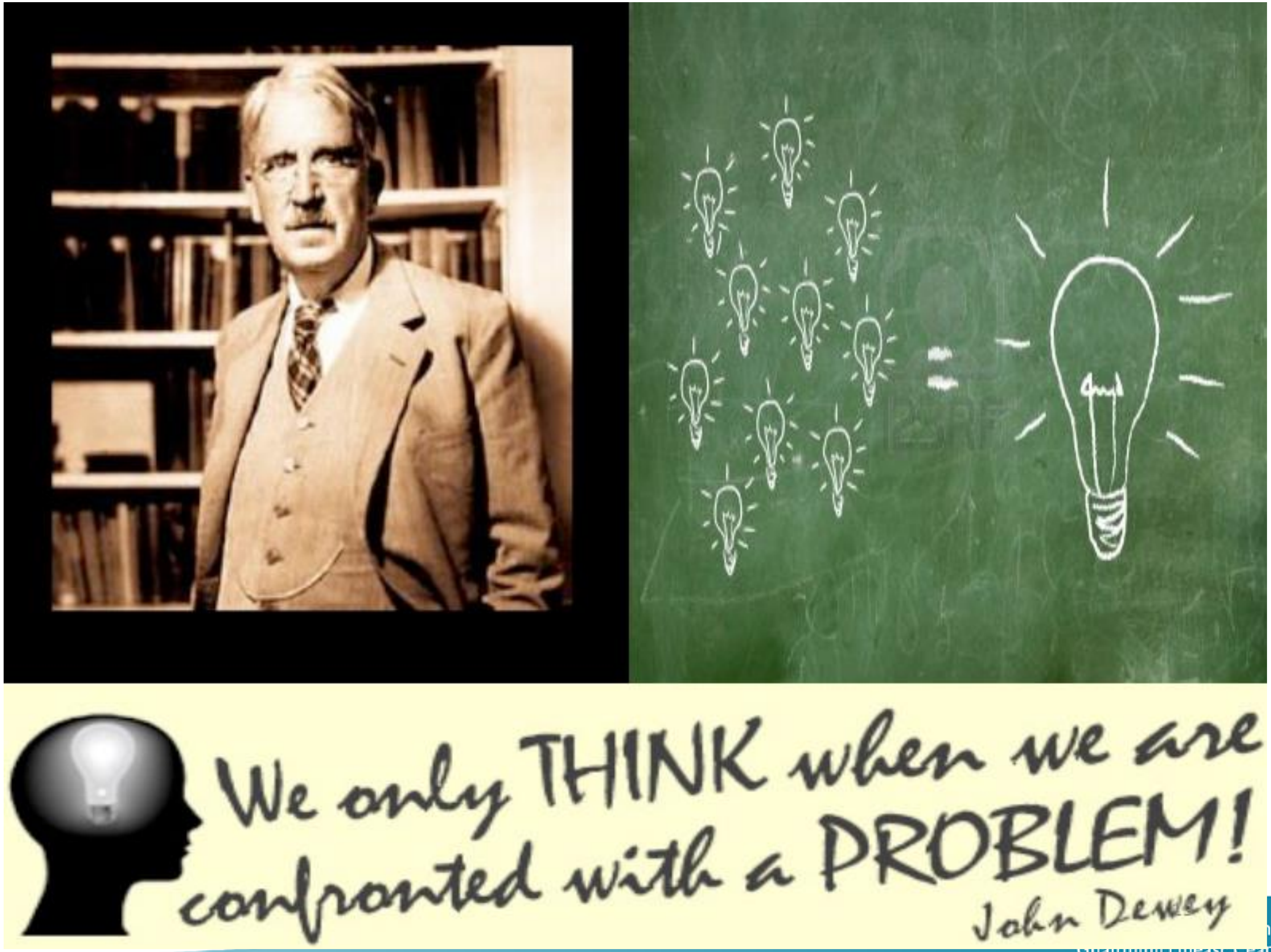
Next Step: Present back to the wider group.

Oide

What CT concepts are you explaining?

What pedagogy are you using?

Presentation


We only THINK when we are confronted with a PROBLEM!
John Dewey

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

LCCS NW2 Session 3

PRIMM
Curriculum planning

By the end of this session…

Participants will be enabled to…

- deepen their understanding of the Investigate, Modify and Make stage of the PRIMM pedagogy by working together through a group activity
- engage collaboratively to develop a curriculum plan for the coming weeks/months guided by the LCCS specification

Successful Strategies and Pedagogies

Oide

Notational Machine
Topic Ordering
Fix the syntax
Reflection
Problem Based Learning
Test Driven Development
Critical Reflection
Code Commenting
Program Tracing / Debugging
Find the 'bug'
Computational Discourse
PRIMM (Use-Modify-Create)
Unplugged Activities
Peer Instruction
Game-based Pedagogy
Pair Programming
Block Programming
Inquiry Based Learning
Fill in the blanks
Turtle Graphics
Parson's Problems
Semantic Waves
Active Learning
Metacognition
Physical Computing

Modelling | Scaffolding | Progression | Context | Constructivism

Example: Fix the syntax

Oide

```
# Run the program to see what happens
# Can you fix the syntax error?
PRINT("Hello World")

# Now continue with the remaining 4 print statements ...
# You will need to uncomment each line and run the program to reveal each
syntax error
#print(Hello World)
#print('Hello World")
#print "Hello World"
#print("Hello", World)
```

Example: Find the bug (semantic error)

Oide

```
# Find and fix the 'bug' in the program below
# The intention is to add a and b and display the answer

a = 3
b = 4
sum = a + 3
print(a, "+", b, "=", sum)
```

Example: Insert comments

Oide

```
# Insert comments to explain each line of code below
# (the first one has been done to get you started)

x = 23 # Assign the value 23 to the variable x
y = 17
print("The value of x is", x)
print("The value of y is", y)
x = x + y
print("The value of x is", x)
x = y
print("The value of x is", x)
```

Example: Fill in the blanks

```python
# A program to demonstrate the multiple if statement
import random

number = random.randint(1, 10)
print(number) # comment this line out later!

guess = int(input("Enter a number between 1 and 10: "))

# Evaluate the condition
if guess == number:
        print("Correct")
        print("Well done!")
elif guess < number:
    print("Hard luck!")
    print("Too low")
else:
    print("Hard luck!")
    print("Too high")

print("Goodbye")
```

```python
if guess > number:

elif guess == number:

else:

```

Example 1: Parson's Problem

Arrange the blocks of code below into the correct order

```
elif guess < number:
    print("Hard luck!")
    print("Too low")
```

```
import random
number = random.randint(1, 10)
```

```
print("Goodbye")
```

```
else:
    print("Hard luck!")
    print("Too high")
```

```
if guess == number:
    print("Correct")
    print("Well done!")
```

```
guess = int(input("Enter a number between 1 and 10: "))
```

The final program should generates a random number, prompts the user to enter a guess and display a message telling the user if the guess was correct, too low or too high.

The program should always display the string *Goodbye* at the end.

Example 1: Parson's Problem

Arrange the blocks of code below into the correct order

4
```
elif guess < number:
    print("Hard luck!")
    print("Too low")
```

1
```
import random
number = random.randint(1, 10)
```

6
```
print("Goodbye")
```

5
```
else:
    print("Hard luck!")
    print("Too high")
```

3
```
if guess == number:
    print("Correct")
    print("Well done!")
```

2
```
guess = int(input("Enter a number between 1 and 10: "))
```

The final program should generate a random number, prompts the user to enter a guess and display a message telling the user if the guess was correct, too low or too high.

The program should always display the string *Goodbye* at the end.

Example 2: Parson’s Problem

Rearrange the jumbled up lines shown below so that the program prompts the end-user to enter two integers and then computes and displays their sum.

```
number2 = int(number2)

number1 = int(input("Enter first number: "))

sum = sum + number1

number1 = int(number1)

print(number1, "+", number2, "=", sum)

number2 = input("Enter second number: ")

print("The answer is sum")

sum = number1 + number2
```

Warning! There are *three* extra lines that you won’t need.

Example 2: Parson's Problem

Rearrange the jumbled up lines shown below so that the program prompts the end-user to enter two integers and then computes and displays their sum.

```
number2 = int(number2)  3
number1 = int(input("Enter first number: "))  1
sum = sum + number1
number1 = int(number1)
print(number1, "+", number2, "=", sum)  5
number2 = input("Enter second number: ")  2
print("The answer is sum")
sum = number1 + number2  4
```

Warning! There are *three* extra lines that you won't need.

Peer Instruction

Oide

Well-evidenced pedagogical strategy

Combination of:

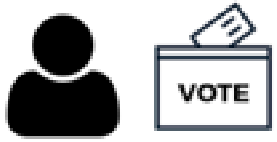
- Flipped learning
- Collaborative working
- Well-chosen MCQs

```
x = 0
y = (x == 21%7)
print(y)
```

A. 0
B. 3
C. False
D. True
E. None of the above

Most effective where there are close distractors and known misconceptions

Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
Students study a topic/concept in advance of the class	Students take a Multiple Choice Quiz in class.	Students discuss their answers in groups	Group decides on a single 'group answer'	Students take part in teacher facilitated discussion



For more information on peer instruction see http://peerinstruction4cs.org

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

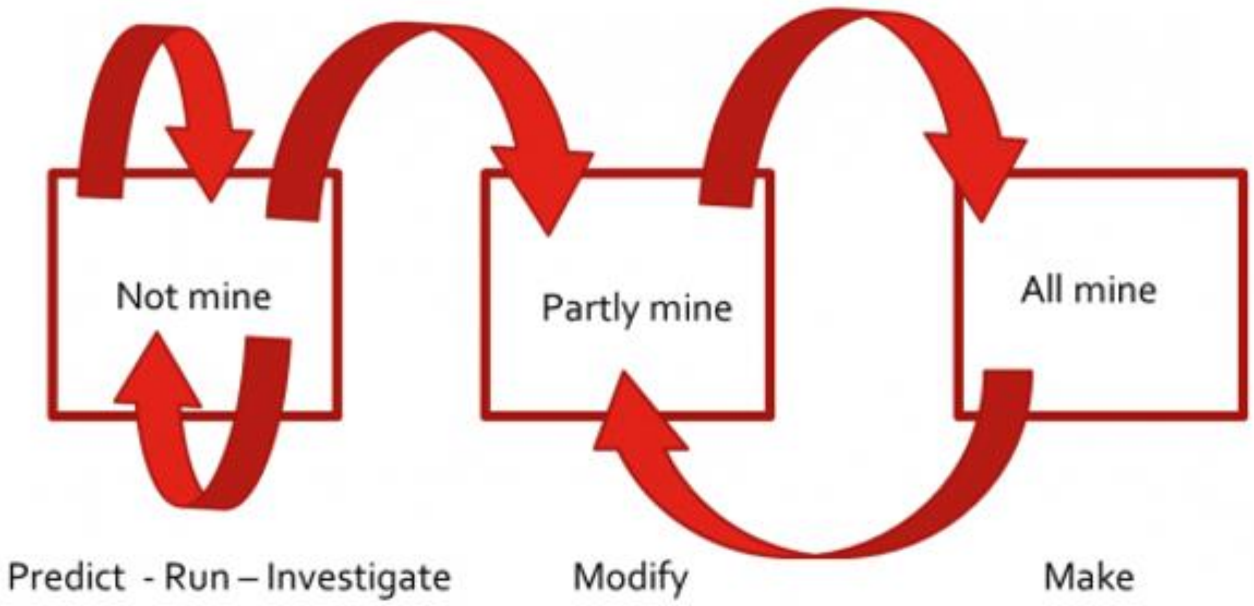
Supporting the Professiona Learning of School Leaders and Teachers

PRIMM

PRIMM

A way of structuring programming lessons that focuses on:

- Reading before Writing
- Student Collaboration
- Reducing Cognitive Load
- Well-chosen starter programs
- Ownership Transfer



Sources:

1. https://blogs.kcl.ac.uk/cser/2017/02/20/exploring-pedagogies-for-teaching-programming-in-school/ (Sue Sentence)
2. https://blogs.kcl.ac.uk/cser/2017/09/01/primm-a-structured-approach-to-teaching-programming/ (Sue Sentence)
3. Sue Sentance, Jane Waite & Maria Kallia (2019) Teaching computer programming with PRIMM: a sociocultural perspective, Computer Science Education, 29:2-3, 136-176, DOI: 10.1080/08993408.2019.1608781

PRIMM

Predict: given a working program, what do you think it will do? (at a high level of abstraction)

Run: run it and test your prediction

Investigate: What does each line of code mean? (get into the nitty gritty - low level of abstraction - trace/annotate/explain/talk about parts)

Modify: edit the program to make it do different things (high and low levels of abstraction)

Make: design a new program that uses the same nitty gritty but that solves a new problem

PRIMM – Example (1 of 2)

Oide

```
import random
number = random.randint(1, 10)
#print(number)
guess = int(input("Enter a number between 1 and 10:"))
if guess == number:
    print("Your guess was correct")
    print("Goodbye")
else:
    print("Incorrect guess")
    print("Goodbye")
```

Predict: Discuss in pairs. What do you think the above program will do? Be precise. Be succinct.

Run: Download the program / Key it in. Execute the program. Test your prediction. Were you correct?

P7

Breakout Activity:

Investigate: Devise some questions to elicit student learning and curiosity. What if … Try … Explain …

Modify: Suggest some simple extensions / modifications for students to make in pairs. Same program.

Make: Formulate new problems that are conceptually similar. New context. New program (copy+paste)

PRIMM – Example (2 of 2)

Oide

```
1.  import random
2.
3.  number = random.randint(1, 10)
4.  #print(number)
5.
6.  guess = int(input("Enter a number between 1 and 10: "))
7.
8.  if guess == number:
9.      print("Your guess was correct")
10.     print("Goodbye")
11. else:
12.     print("Incorrect guess")
13.     print("Goodbye")
```

Investigate:

1. Uncomment line 4. What happens?
2. What is the purpose of line 4?
3. What would happen if you removed `int` from line 6
4. Try changing `==` to `!=` on line 8. What happens?
5. What if `==` was changed to `=` ?
6. What would happen if you don't enter an integer?
7. Try removing a bracket (anywhere). What happens
8. Annotate each line of the program.

Modify:

1. Change the program so that it generates a number between 1 and 100? Can you be sure?
2. Change the program so that there is only one `print("Goodbye")` statement (without altering the logic)
3. Extend the program so that it tells the user if the number entered was *too high* or *too low*
4. Design an algorithm based on the program that would give the user 3 guesses
5. Get the computer to generate 4 numbers (lotto) OR ask the user how many numbers to generate?

Make:

Write a program that generates two numbers and prompts the user to enter their product

Group activity

Instructions:

In your groups, fill in the *Investigate*, *Modify* and *Make* sections in your workbook for the code snippet assigned to you.

You may use the examples from the previous pages to help you.

P8-15

Task 1

```
1. from turtle import *
2.
3. color("red")
4. pensize(5)

5. forward(100)
6. left(90)
7. forward(100)
8. left(90)
9. forward(100)
10.left(90)
11.forward(100)
```

Task 2

```
1. centigrade = float(input("Enter the Centigrade value: "))
2. fahrenheit = 9/5 * centigrade + 32
3. print(centigrade, "degrees C equals", fahrenheit, "degrees F")
```

Task 3

```
runningTotal = 0

price1 = 10
runningTotal = runningTotal + price1
price2 = 14
runningTotal = runningTotal + price2
price3 = 6
runningTotal = runningTotal + price3

print("Total amount is", runningTotal)
```

Task 4

```
print("Average height calculator")
print("=========================")

h1 = int(input("Enter first height (cm): "))
h2 = int(input("Enter second height (cm): "))
h3 = int(input("Enter third height (cm): "))
h4 = int(input("Enter fourth height (cm): "))
h5 = int(input("Enter fifth height (cm): "))

avgHeigth = (h1+h2+h3+h4+h5)/5

print("The average height is ", avgHeigth, "cm")
```

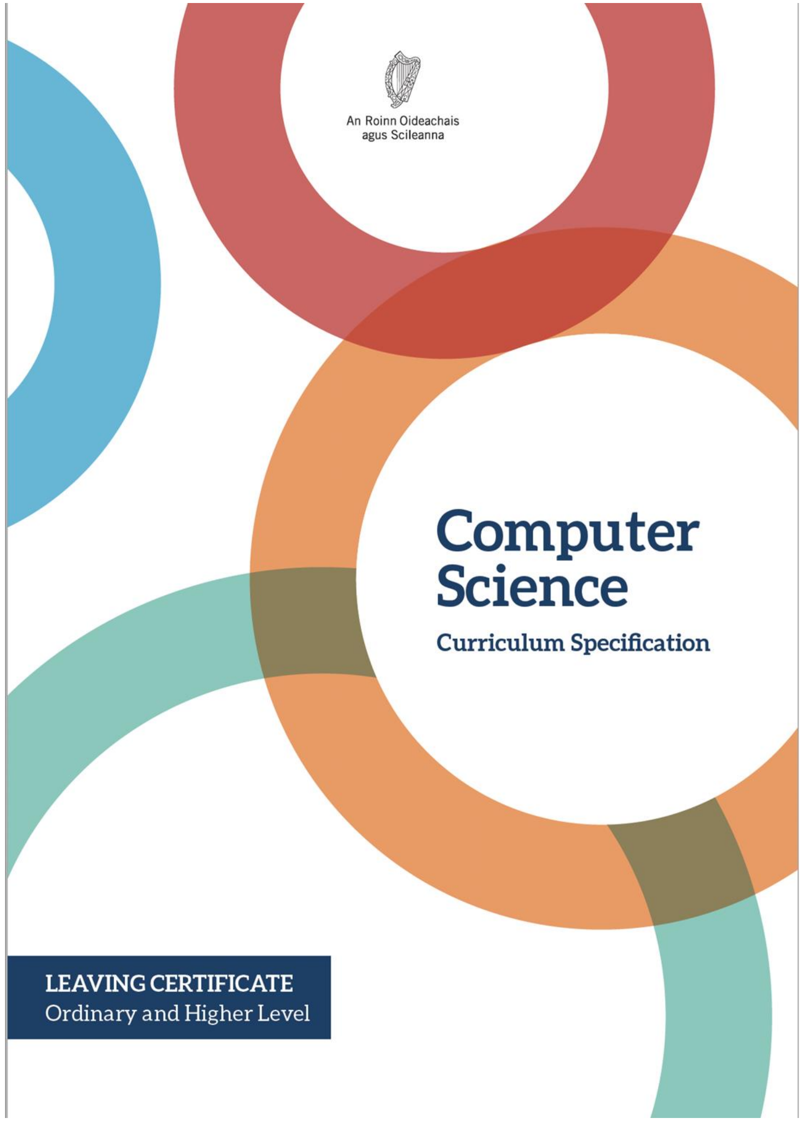
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Curriculum planning

Oide

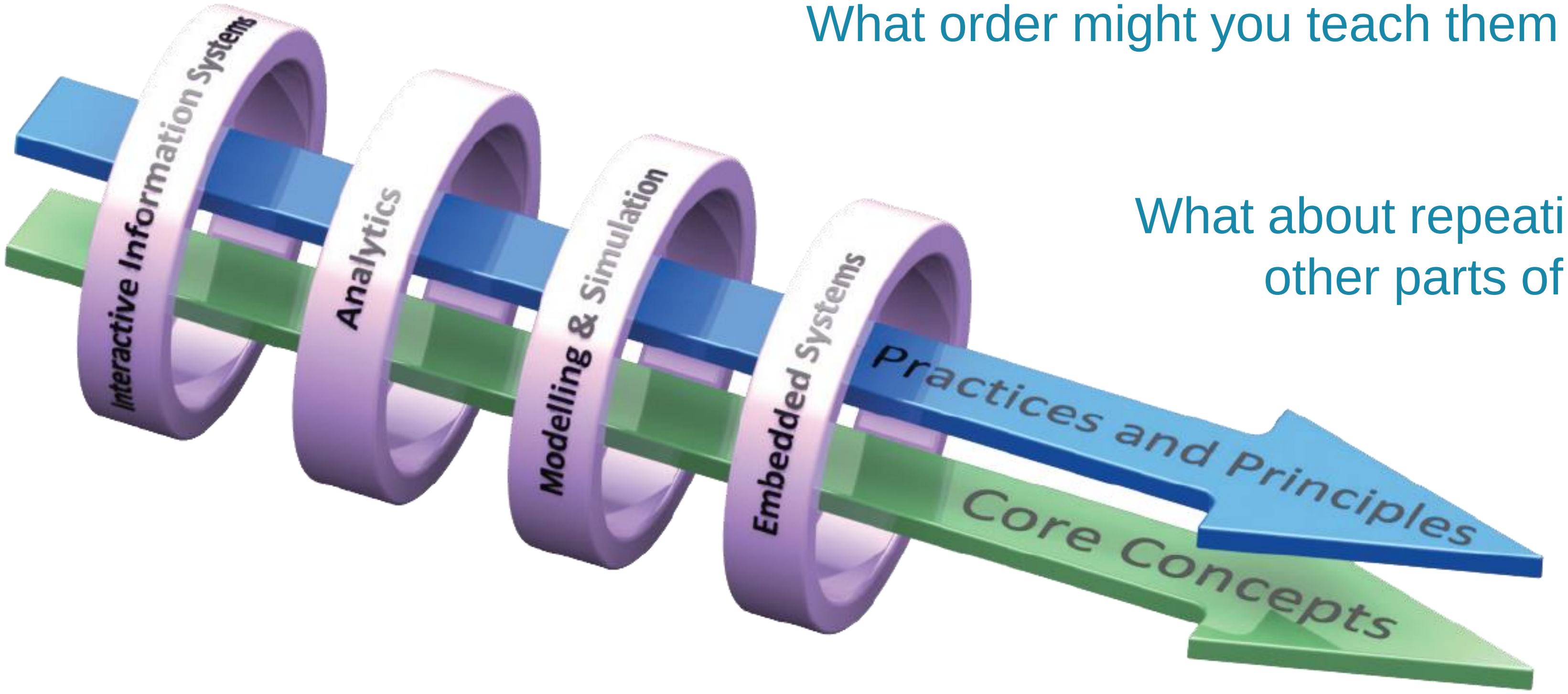


> *"Learning outcomes can best be defined as statements of what a learner knows, understands and is able to do after completion of learning."*
>
> CEDEFOP (2009)

How might you work with the learning outcomes?

What order might you teach them in?

What about repeating LOs / linking to other parts of the course?



How might students demonstrate they have achieved the learning outcomes?

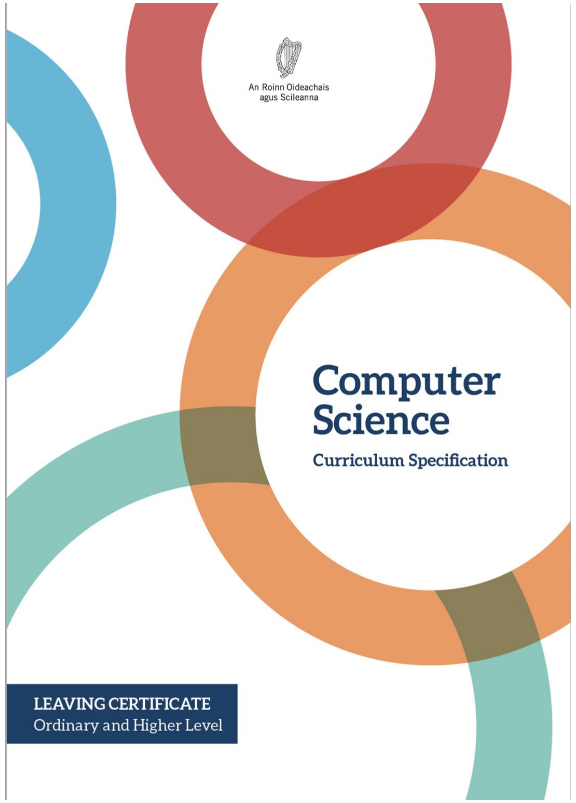
What content or resources might you need?

Group activity

Use the LCCS specification, consider the following question:
How do you intend to approach LCCS in your classroom (over the next 4 weeks/until mid-term/until Christmas)?

In your groups, consider:
Timeframe / Topics / LOs / Resources / Assessment / Build up to ALTs / ALTs / Equipment etc.

Nominate:
A **notetaker** to summarise your group’s work
A **spokesperson** to provide feedback



Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

Leaving Certificate Computer Science

National Workshop 2

Day 2

Day 2 - Workshop Overview

Session 4 09:00 - 11:00	Introduction to ALT4
Tea/Coffee 11:00 - 11:15	
Session 5 11:30 - 13:00	ALT4: Investigate + Plan
Lunch 13:00 - 14:00	
Session 6 14:00 - 15:30	ALT4: Design + Create

Key Messages

There are many ways to use the LCCS specification

The Applied Learning Tasks (ALTs) provide an opportunity to teach theoretical aspects of LCCS

The learning outcomes (LOs) are non-linear

LCCS can be mediated through a constructivist pedagogical approach

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

LCCS NW2 Session 4

Introduction to ALT4

By the end of this session…

Participants will …

- be introduced to ALTs
- be introduced to ALT4
- develop an understanding of Embedded systems
- be introduced to Micro:bit – Demonstration
- participate in Micro:bit group activities
- develop an understanding of Design Methodologies

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Introduction to ALTs

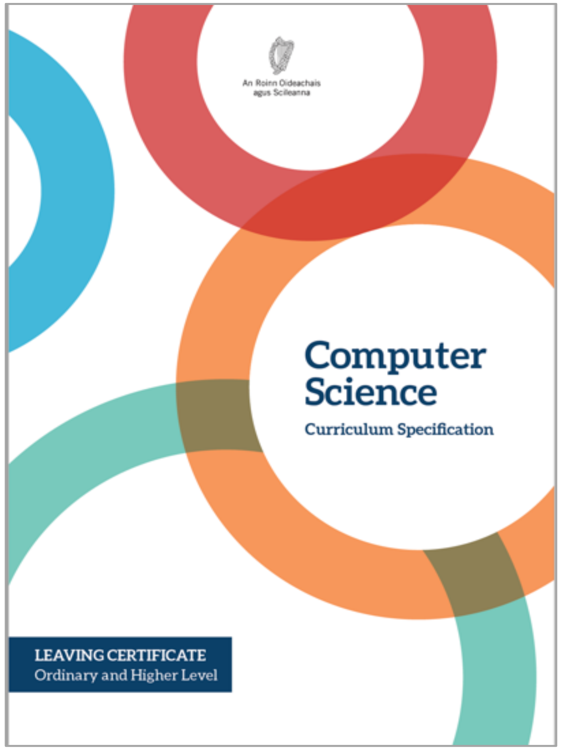
Applied Learning Tasks (ALTs)

Students work in teams to carry out four applied learning tasks over the duration of the course each of which results in the creation of a real or virtual computational artefact and a report.

These artefacts should relate to the students' lives and interests.

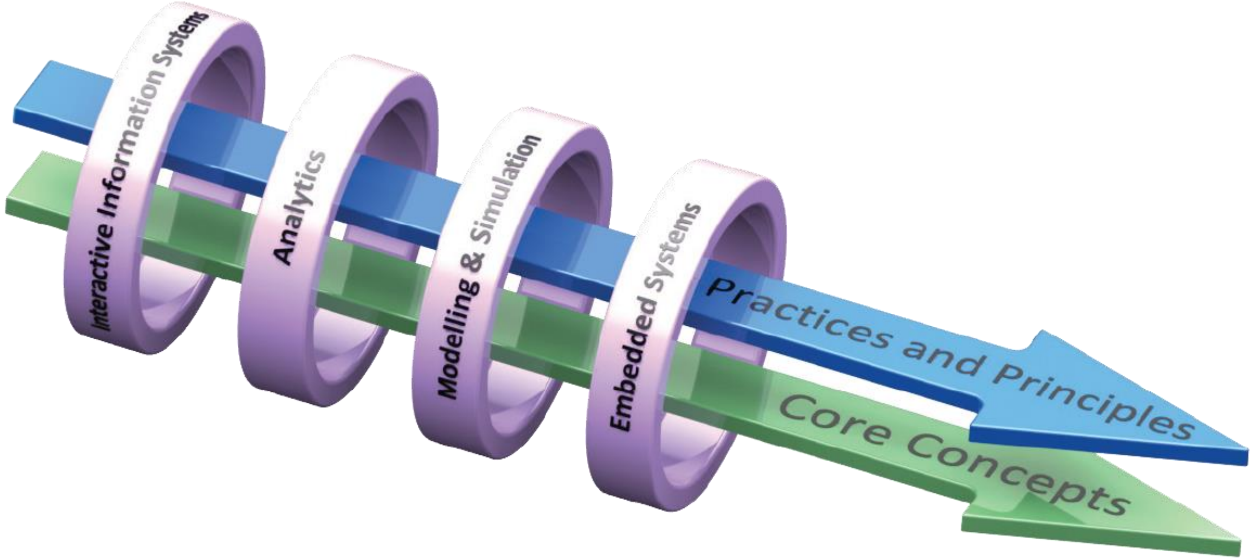
Where possible, the artefacts should be beneficial to the community and society in general.

Examples of computational artefacts include programs, games, web pages, simulations, visualisations, digital animations, robotic systems, and apps.



LCCS Specification page 15

LCCS Interwoven



The four applied learning tasks explore the four following contexts:

1 - Interactive information systems
2 - Analytics
3 - Modelling and simulation
4 - Embedded systems

Key point to remember: Explore and teach the LOs through the lens of ALTs.

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Introduction to ALT4

Considering the ALTs…

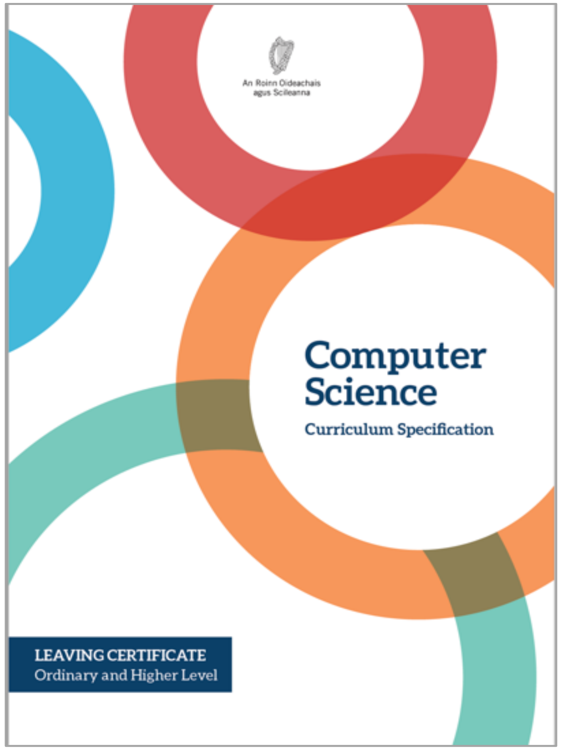
Strand 1: Practices and principles	Strand 2: Core concepts	Strand 3: Computer science in practice
Computers and society	Abstraction	Applied learning task 1 - Interactive information systems
Computational thinking	Algorithms	Applied learning task 2 - Analytics
Design and development	Computer systems	Applied learning task 3 - Modelling and simulation
	Data	Applied learning task 4 - Embedded systems
	Evaluation/Testing	

ALT4 - Embedded systems

The design and application of computer hardware and software are a central part of computer science.

Students will implement a microprocessor system that uses sensors and controls digital inputs and outputs as part of an embedded system.

By building the component parts of a computer system, students will deepen their understanding of how computers work and how they can be embedded in our everyday environments.



LCCS Specification
page 23

Oide

ALT4 - Learning outcomes

Students learn about:	Students should be able to:
Embedded systems	3.11 use and control digital inputs and outputs within an embedded system
Computing inputs and outputs	3.12 measure and store data returned from an analogue input
Computer systems	3.13 develop a program that utilises digital and analogue inputs
Design process	3.14 design automated applications using embedded systems

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Embedded Systems

Activity: Think-Pair-Share

Oide

Participants spend time in silence writing or thinking about their own ideas

Participants turn to the person beside them to discuss their ideas

Pairs share their answers with other pairs (square) or the wider group

Consider and discuss:

1. What are the uses of Embedded Systems?

2. What is the difference between digital and analogue data?

P34

Embedded Systems

General Purpose PC

Embedded System

P35

This washing machine has an embedded system in it.

The microcontroller displays the status of the machine in the display

You can program the microcontroller by pressing the buttons and turning the dial

The microcontroller in this embedded system controllers the speed of the motor (drum)

More like Embedded System

More like PC

In a car, we have a lot of embedded systems as shown below. We will have more and more as cars evolve into future car.

https://www.sharetechnote.com/html/EmbeddedSystem_WhatIsIt.html

Embedded Systems

Embedded systems are a combination of hardware and software designed to perform a specific function. They are called ‘embedded’ because they are often used as part of a larger system. Many embedded systems use sensors to receive analogue or digital inputs. The input data which is often supplied in real time is then processed resulting in some sort of output. While not every embedded system will have a user interface, some are designed to meet the principles of universal design.

Q15, LCCS HL 2021

Characteristics of an Embedded System:

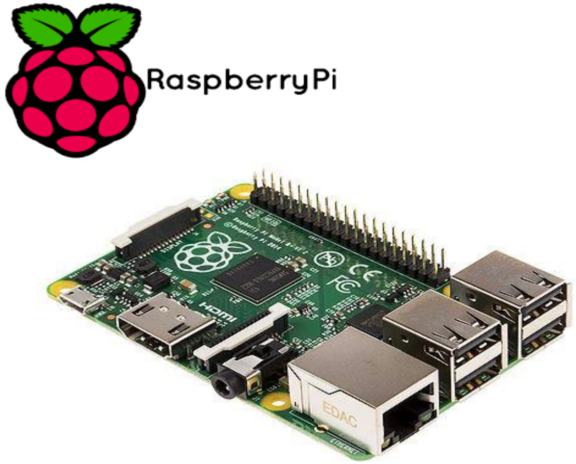
- Task-specific.
- Typically, consists of hardware, software, and firmware;
- Microprocessor-based or microcontroller-based
- Often used for sensing and real-time computing in **Internet of Things (IoT)** devices

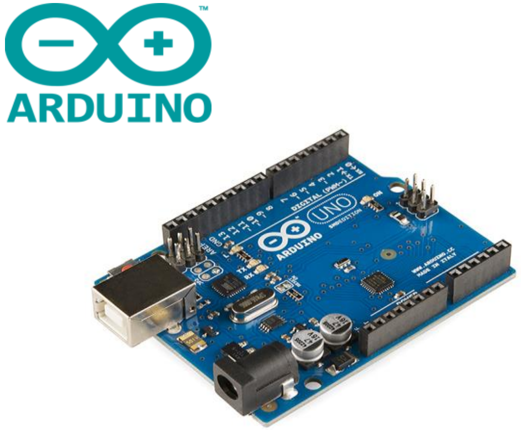
P35

Matching Exercise

P36

Microprocessors/ Microcontrollers





P37

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

Introduction to Micro:bit

Links between Micro:bit and Core Concepts

"The core concepts are developed theoretically and applied practically. In this way, conceptual classroom-based learning is intertwined with experimental computer lab-based learning throughout the two years of the course." PAGE 20 Spec

Strand 1: Practices and principles	Strand 2: Core concepts	Strand 3: Computer science in practice
Computers and society	Abstraction	Applied learning task 1 - Interactive information systems
Computational thinking	Algorithms	Applied learning task 2 - Analytics
Design and development	Computer systems	Applied learning task 3 - Modelling and simulation
	Data	Applied learning task 4 - Embedded systems
	Evaluation/Testing	

LCCS Learning Outcomes

2.11 describe the different components within a computer and the function of those components

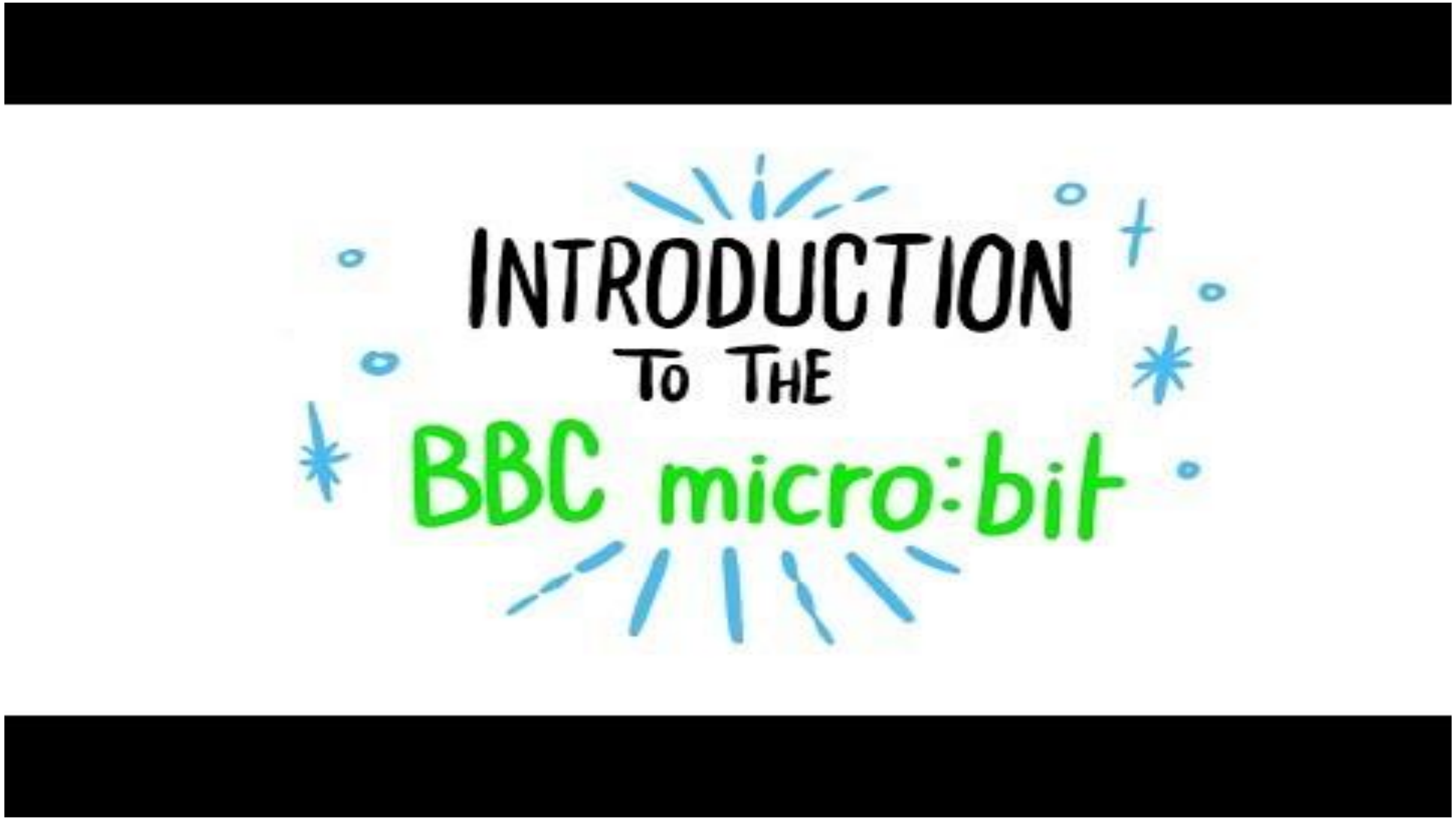
2.12 describe the different types of logic gates and explain how they can be arranged into larger units to perform more complex tasks

2.13 describe the rationale for using the binary number system in digital computing and how to convert between binary, hexadecimal and decimal

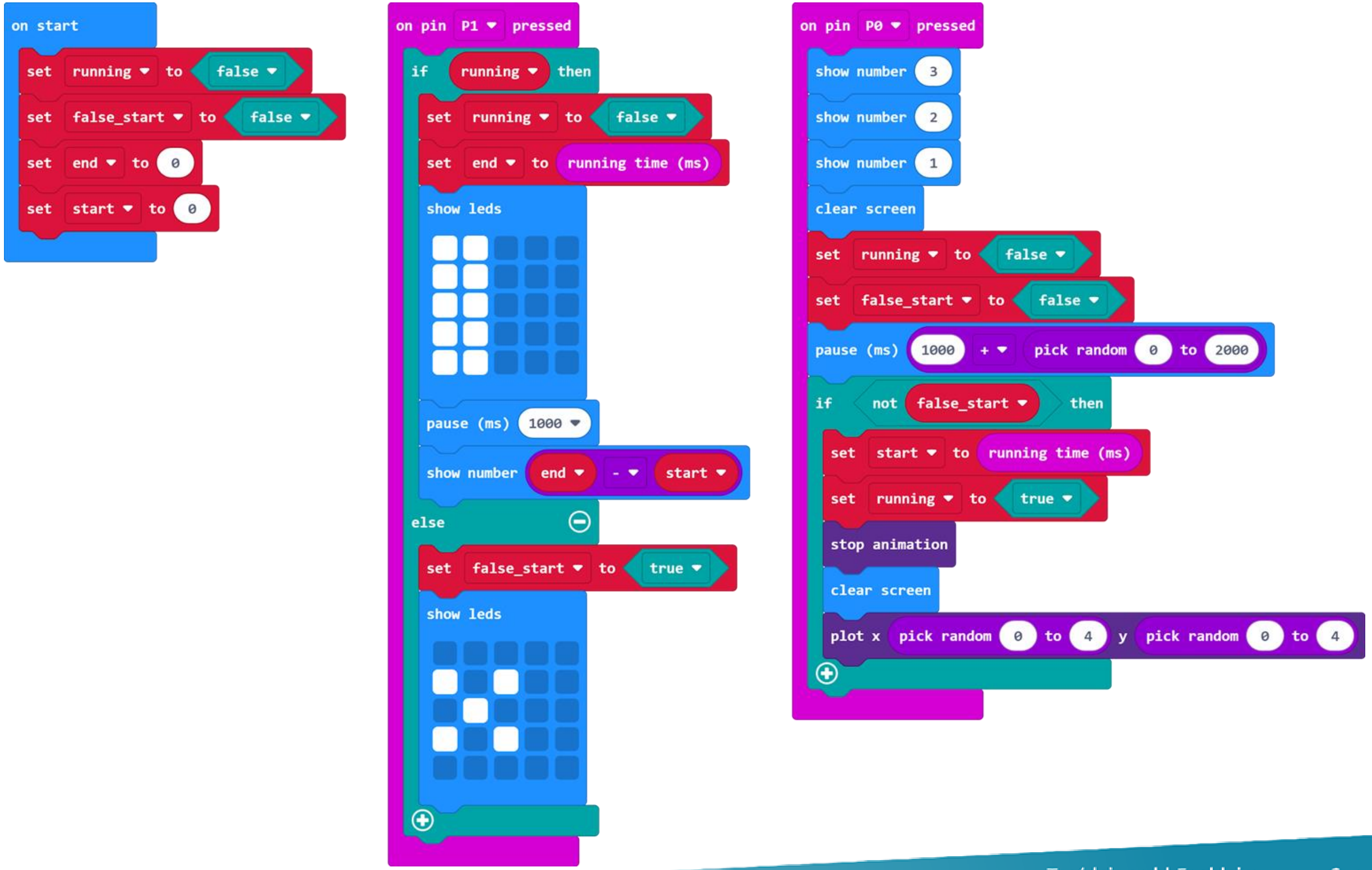
2.14 describe the difference between digital and analogue input

2.15 explain what is meant by the World Wide Web (WWW) and the Internet, including the client server model, hardware components and communication protocols

Getting started



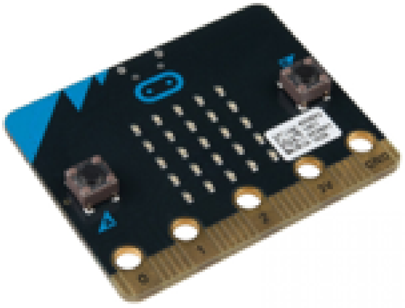
Reaction Game - Demonstration



Resources for Micro:bit

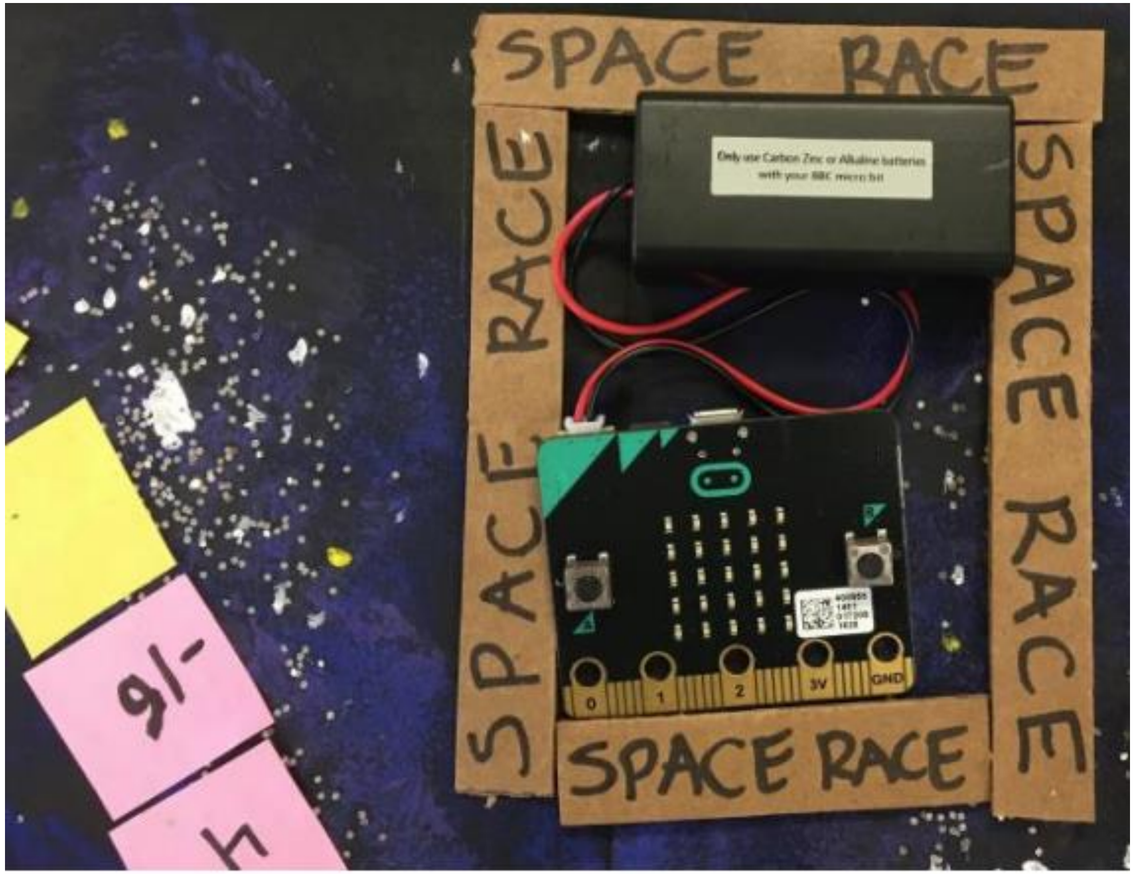
Teaching Programming

using the BBC micro:bit



https://drive.google.com/file/d/1iZ6I3rRvqeUAliAYfWn9mycN--uIONJy/view?usp=sharing

Resources for Micro:bit



https://makecode.microbit.org/courses/csintro

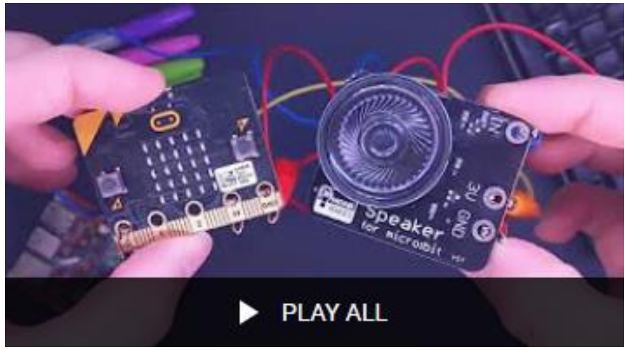
Lessons

1. Making
2. Algorithms
3. Variables
4. Conditionals
5. Iteration
6. Review/Mini-Project
7. Coordinate grid system
8. Booleans
9. Bits, bytes, and binary
10. Radio
11. Arrays
12. Independent final project

Resources for Micro:bit

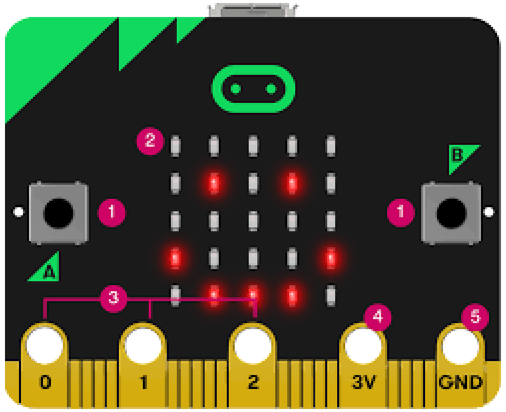
micro:bit classroom — Beta

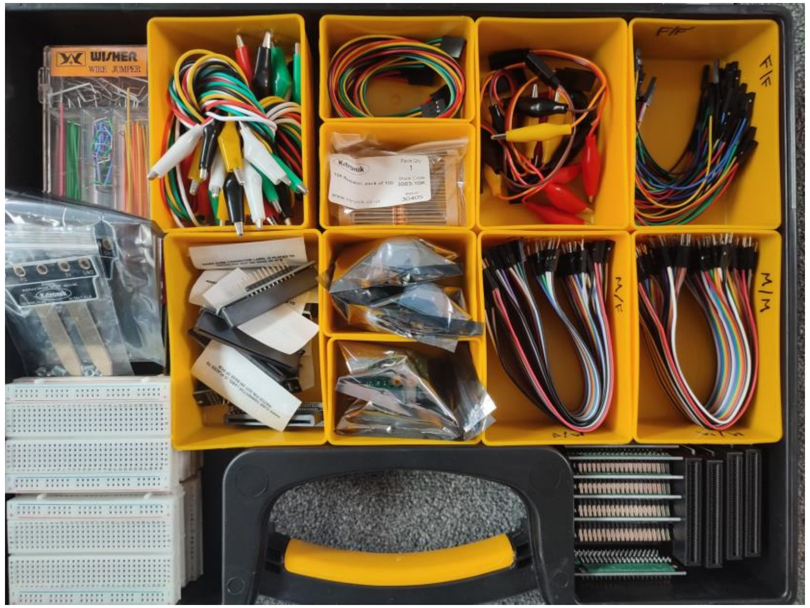
https://classroom.microbit.org/



https://www.youtube.com/playlist?list=PL_0n6wprxG5IaRaYwK3JxxLSPsI33OYSF

Micro:bit kits





Oide

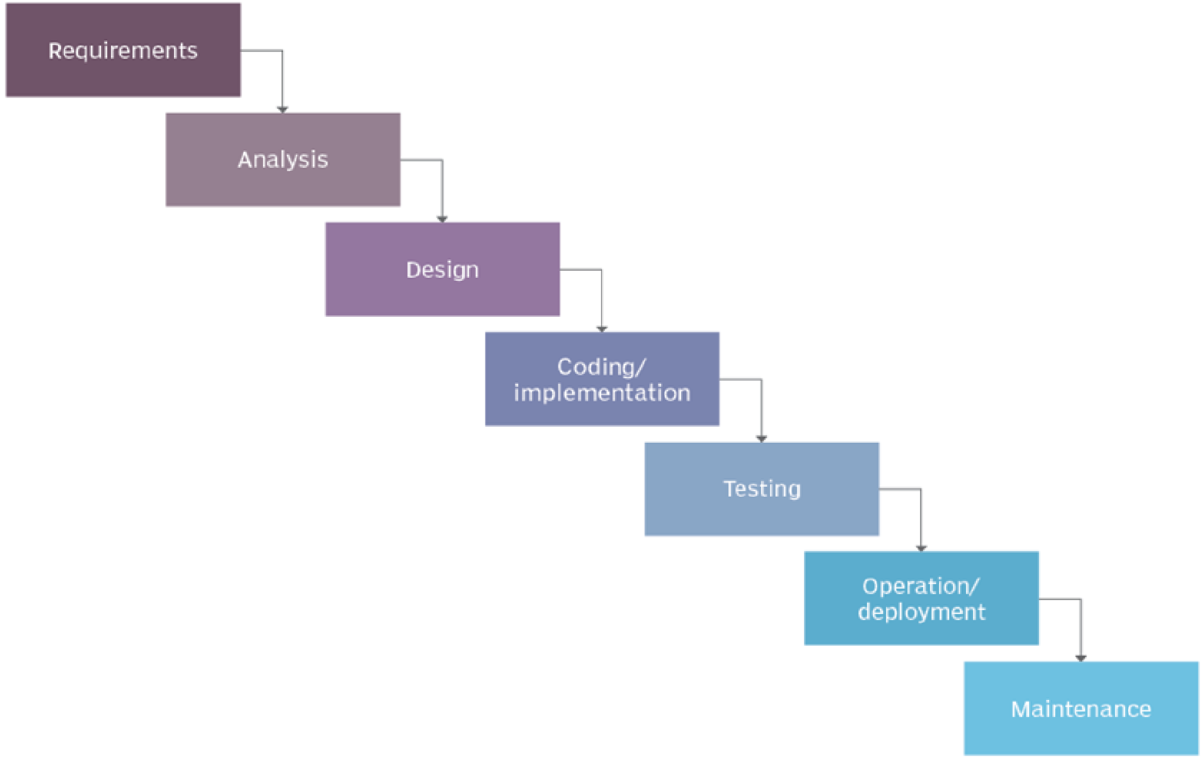
Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

Design methodologies

Agile vs Waterfall

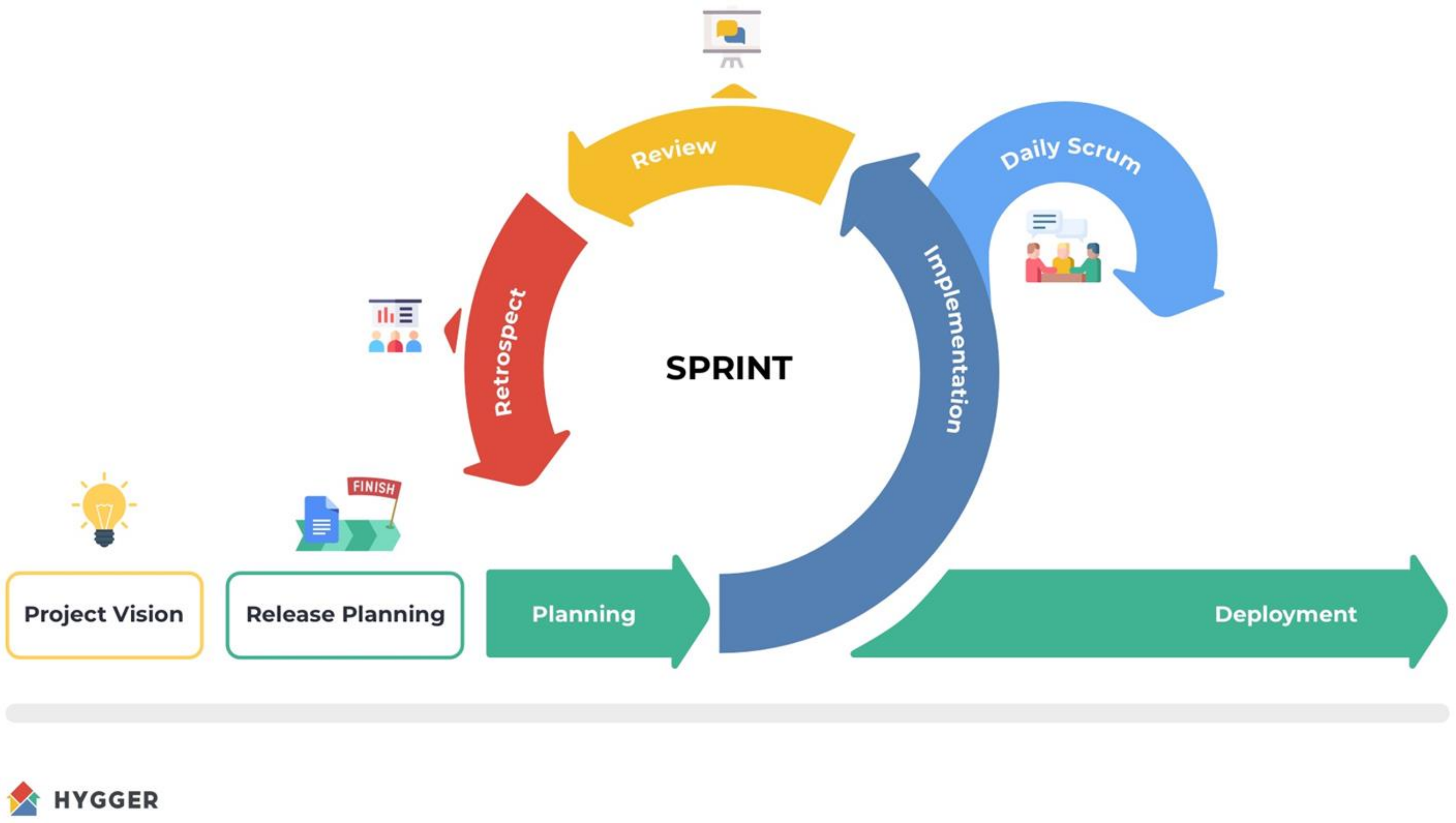
Waterfall



P48

https://www.theserverside.com/tip/Agile-vs-Waterfall-Whats-the-difference

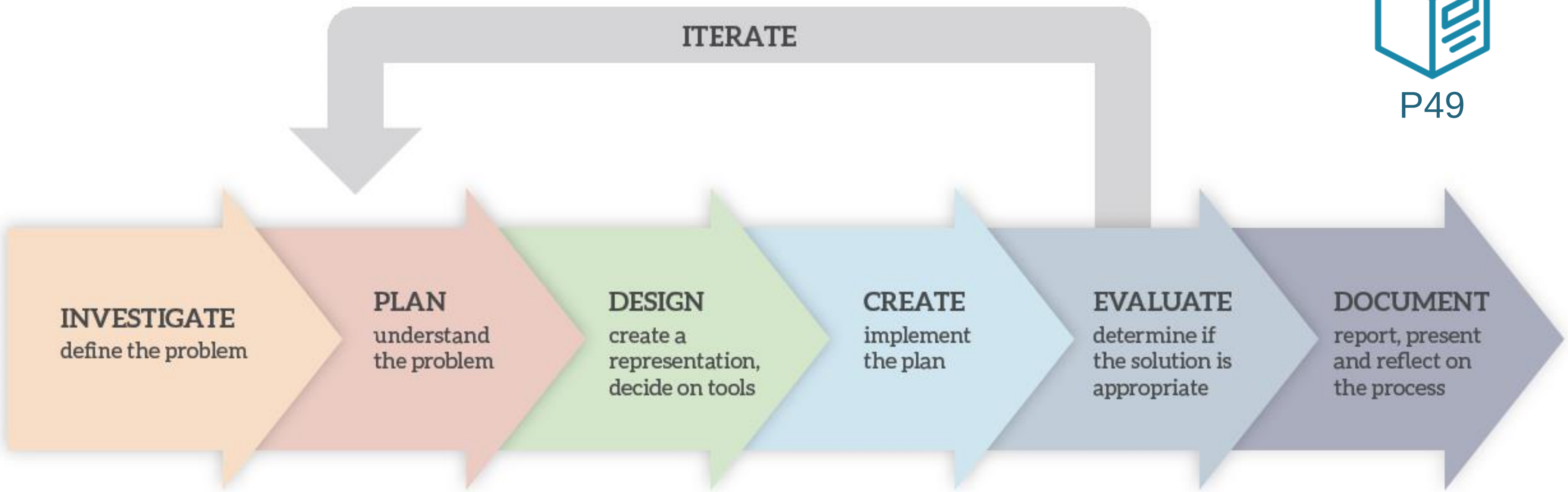
Agile



P48

https://hygger.io/guides/agile/

The Design Process



P49

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

LCCS NW2 Session 5

ALT4:
Investigate and Plan

By the end of this session…

Participants will be enabled to…

- work in groups to share and evaluate potential ideas for ALT 4 (embedded systems)
- collaborate on developing one potential idea for ALT 4 further
- give and receive feedback on potential ALT 4 ideas
- enhance their understanding of the Investigate and Plan stages of the Design Process with a particular focus on ALT 4

The Design Process

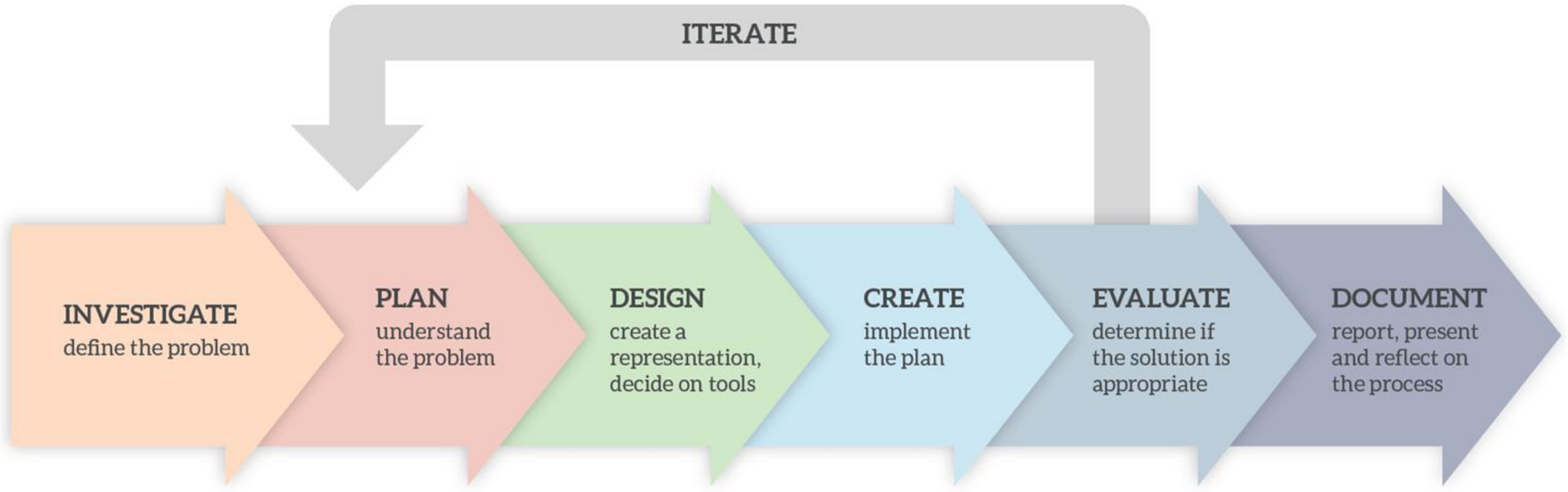


Figure 3: Overview of a design process

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Investigate

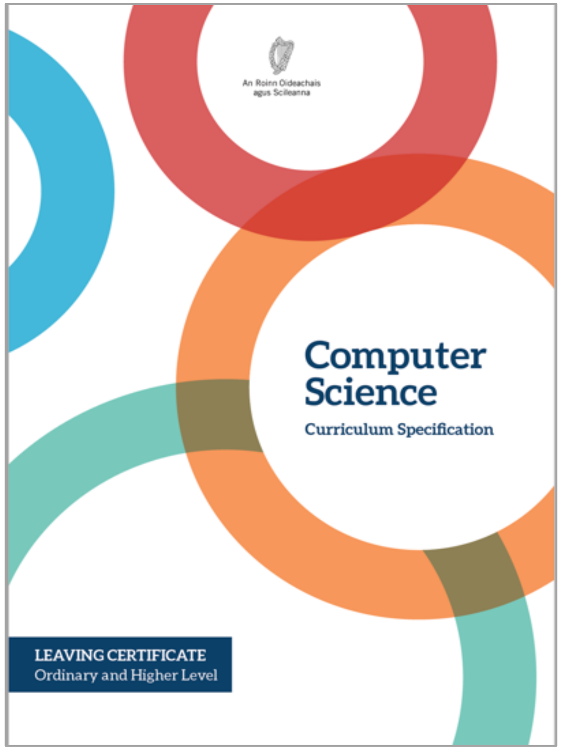
The Design Process: Investigate

ALT4 - Embedded systems

The design and application of computer hardware and software are a central part of computer science.

In this Applied Learning Task, students will implement a microprocessor system that uses sensors and controls digital inputs and outputs as part of an embedded system.

By building the component parts of a computer system, students will deepen their understanding of how computers work and how they can be embedded in our everyday environments.



LCCS Specification
page 23

ALT4 - Learning outcomes

Students learn about:	Students should be able to:
Embedded systems	3.11 use and control digital inputs and outputs within an embedded system
Computing inputs and outputs	3.12 measure and store data returned from an analogue input
Computer systems	3.13 develop a program that utilises digital and analogue inputs
Design process	3.14 design automated applications using embedded systems

ALT4 example: Inuit children

System for Inuit children

LED built into hoods to flash when light is low

Built-in heating system with sensors in positions

ALT4: Investigate

What is an embedded system? Give examples from the world around us.

What are sensors? Digital inputs/outputs? Analogue inputs/outputs?

What are your hobbies/interests/passions? Can you think of example embedded systems that might support these?

What about other examples for users other than yourself e.g. family members, friends, school, community organisation, society?

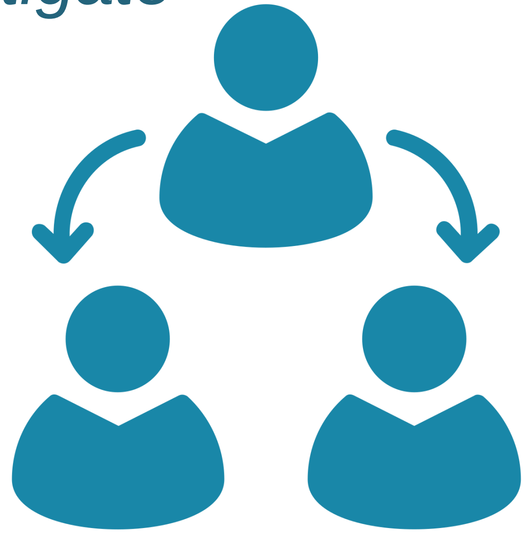
Group activity

In your assigned groups, start brainstorming possible project ideas for students for ALT4

Aim for as many ideas as you can

Record your ideas in your booklet under *ALT 4:Investigate*

P



Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

Plan

The Design Process

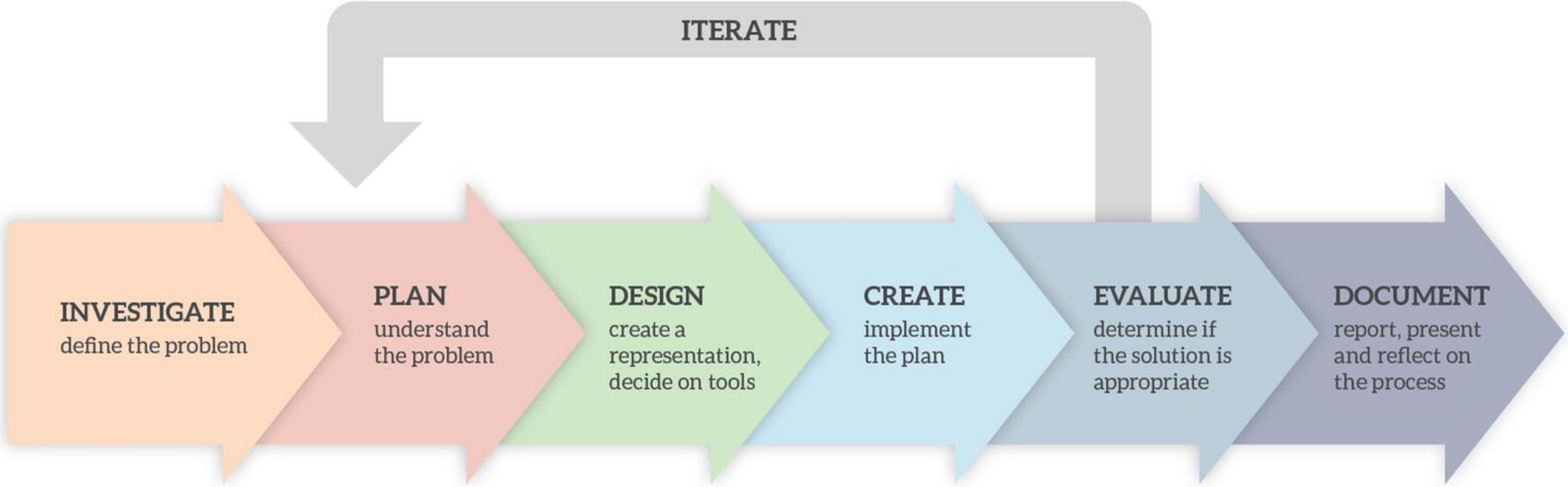


Figure 3: Overview of a design process

ALT4: Plan

Oide

In your assigned groups, **evaluate** your potential ideas for ALT 4

Choose **one idea** for further development

Dissect the idea

You may use the **prompt questions** to help you

The Design Process: Plan

ALT4: Plan

Oide

Is there a broad theme or a specific topic?
Who is the audience?
What teaching & learning strategies could you use?
What does your project do?
Does your project idea cover all the LOs for this ALT?
What other LOs can be taught through the lens of this project?
What tools or materials are needed?
What are the roles in the group?
What research or upskilling do you need to do?

P51

Group activity - Feedback

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

LCCS NW2 Session 6

ALT4: Design and Create

By the end of this session…

Participants will be enabled to…

- enhance their understanding of the Design stage through considering representations and design tools, e.g. Flowcharts

- enhance their understanding of the Create stage of the Design Process

The Design Process

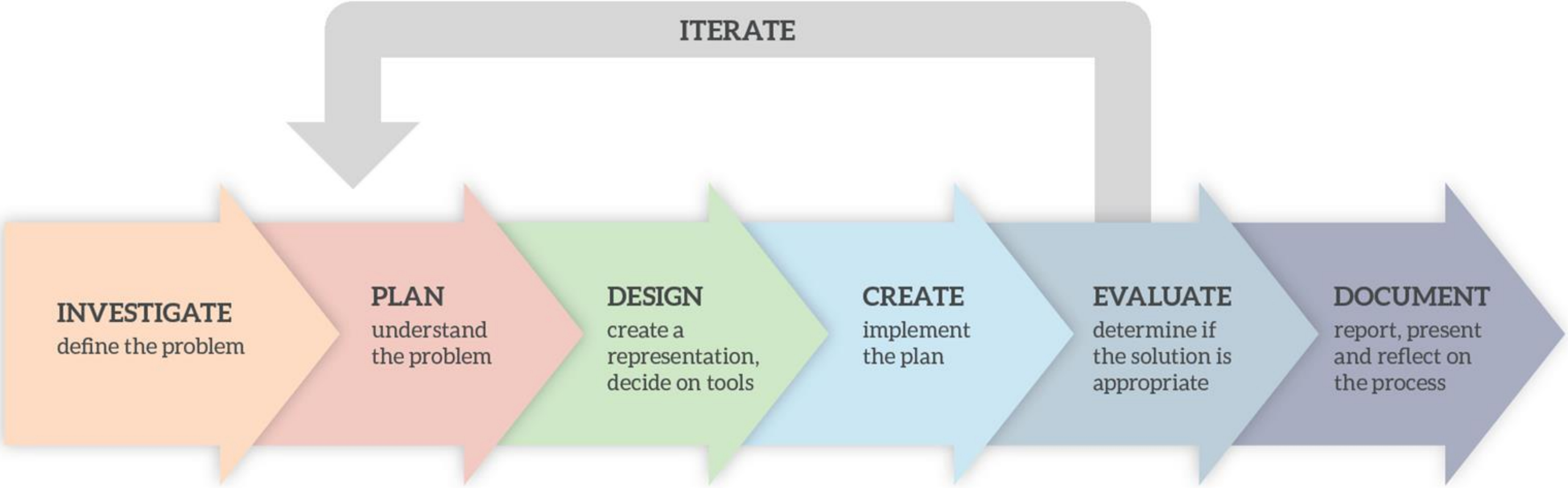
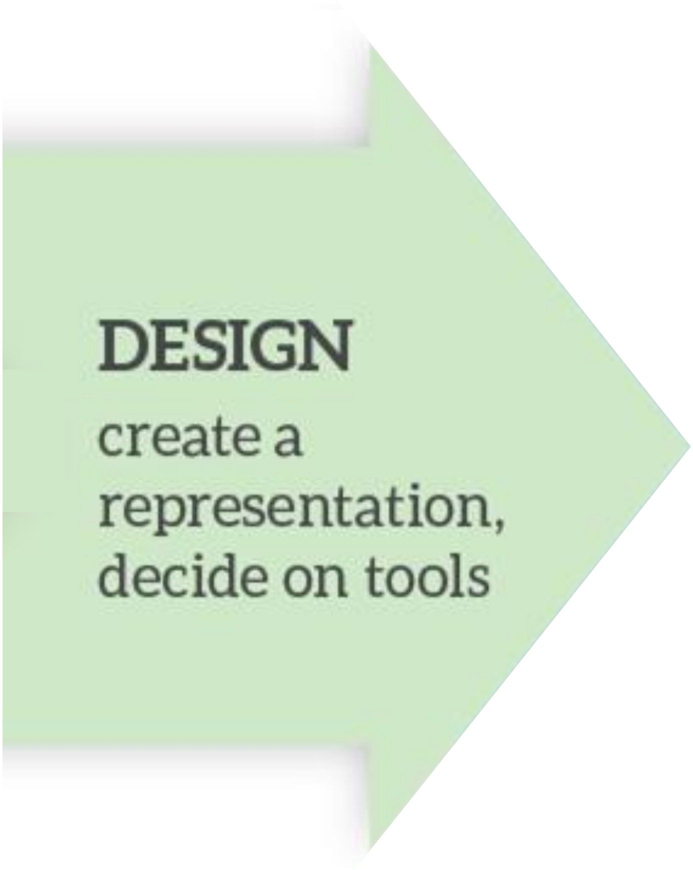


Figure 3: Overview of a design process

The Design Process



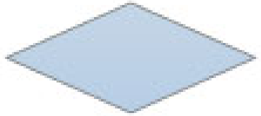
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

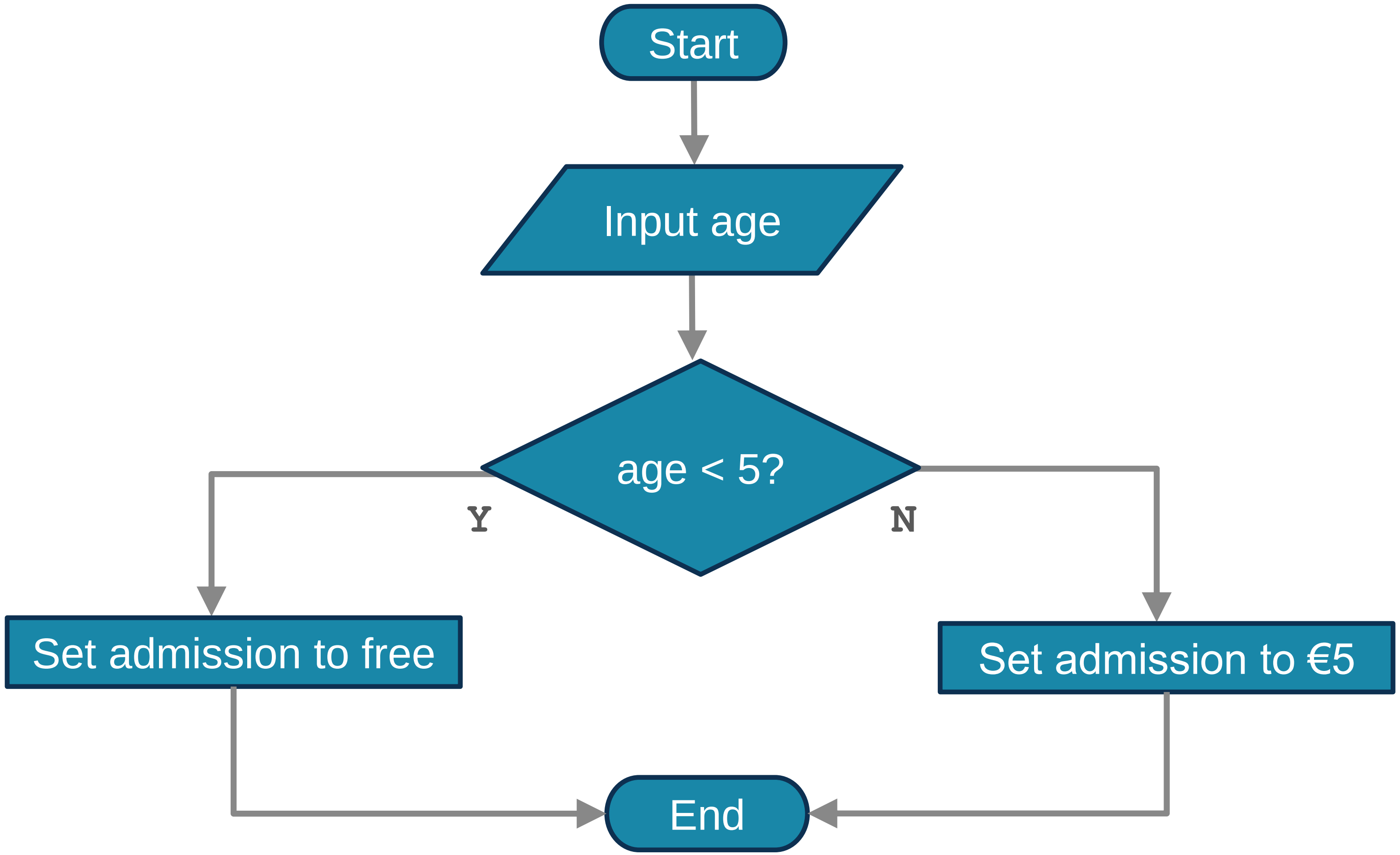
Design

Flow charts

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

P52

Admission example



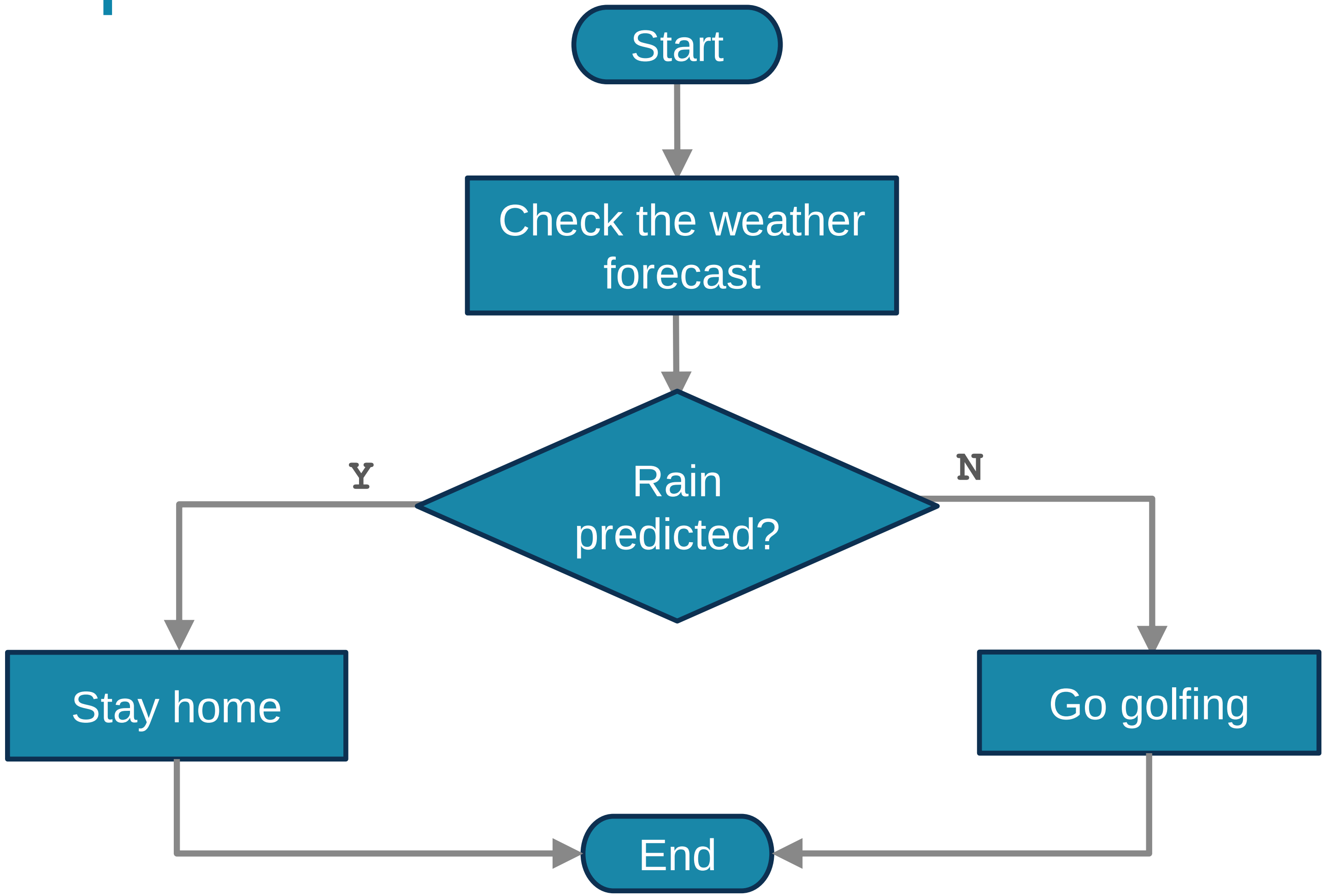
Oide

Pseudocode

```
program start

check weather forecast

if rain predicted
    Stay home
else
    Go golfing
end if

program end
```

P53

Golf example



Group activity

micro:bit

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

The Design Process

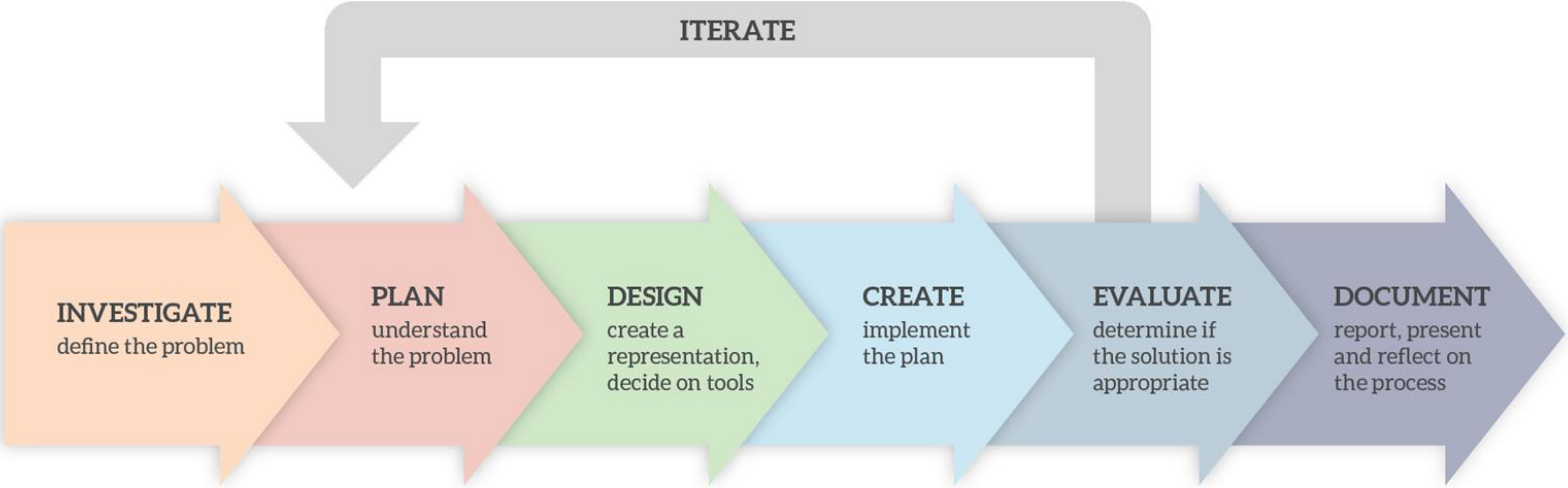
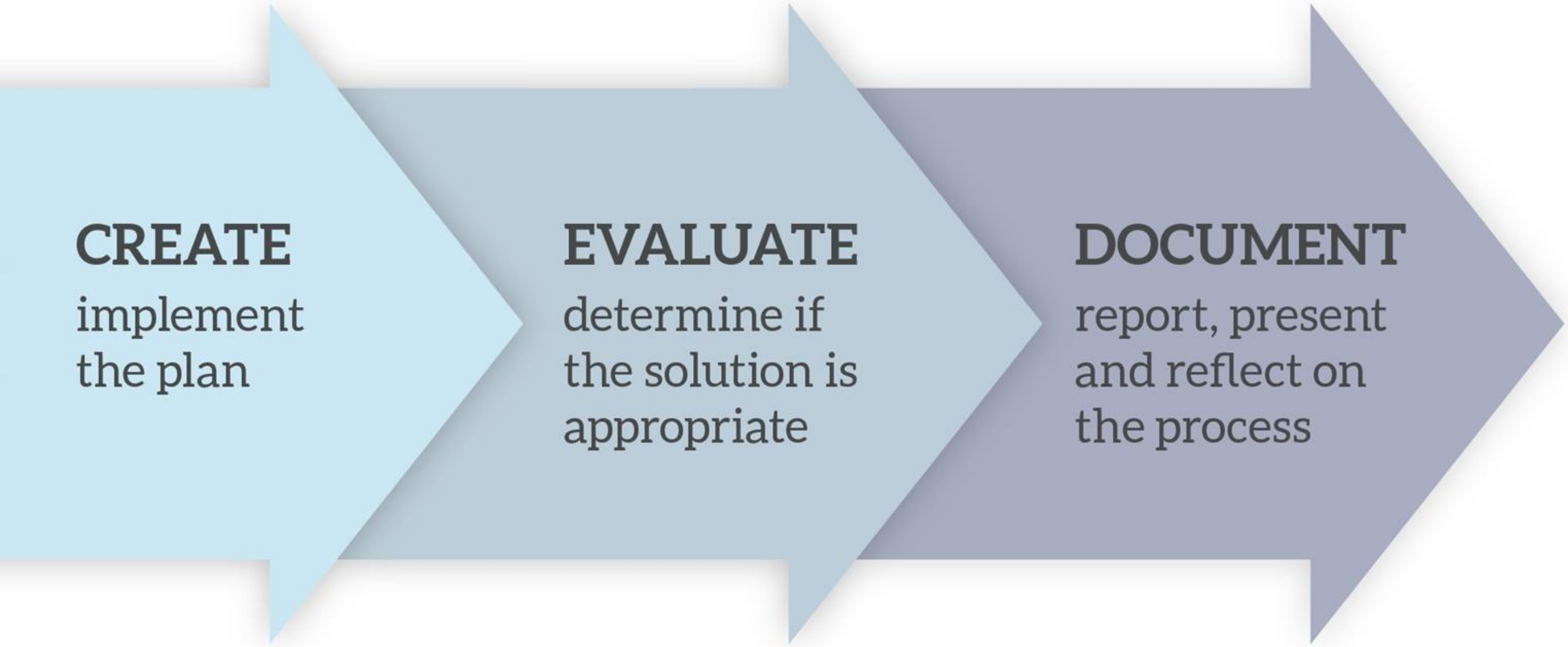


Figure 3: Overview of a design process

Create Evaluate Document



From the Specification

The output from each task is a computational artefact and a concise individual report outlining its development.

In the report, students outline where and how the core concepts were employed.

The structure of the reports should reflect the design process shown above in Figure 3.

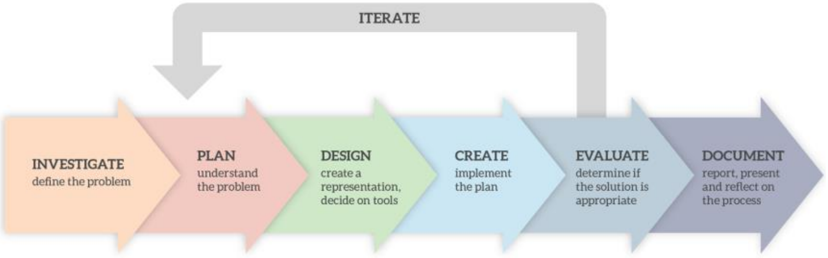
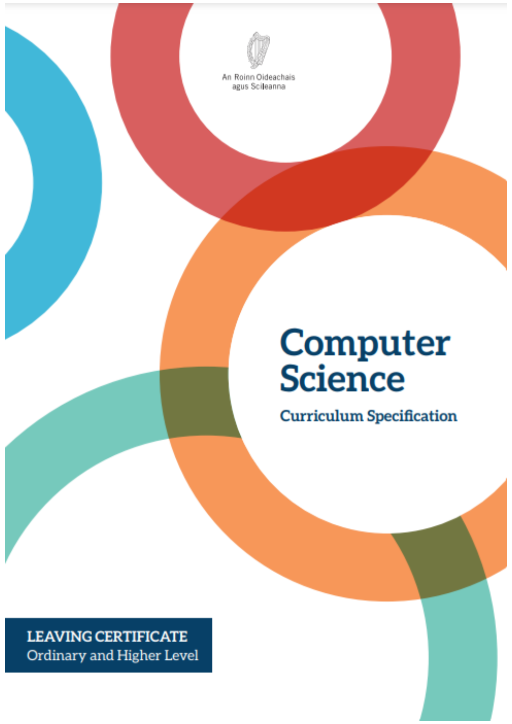


Figure 3: Overview of a design process



Page 11

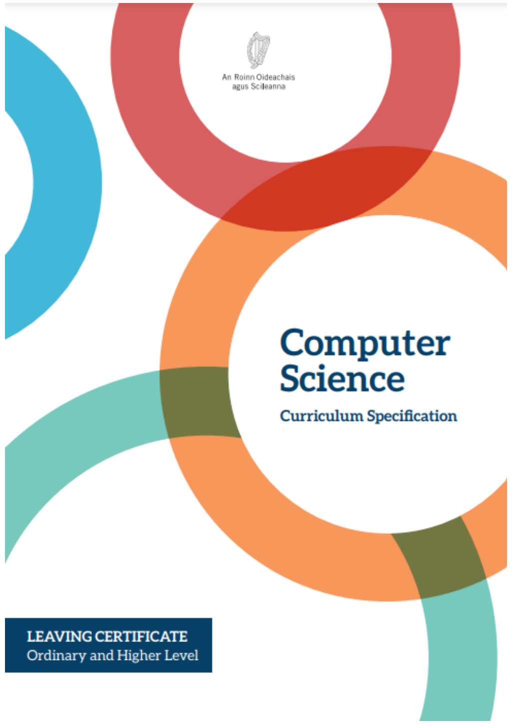
Oide

From the Specification

Initial reports could be in the form of structured presentations to the whole class.

As students progress, reports should become detailed and individual.

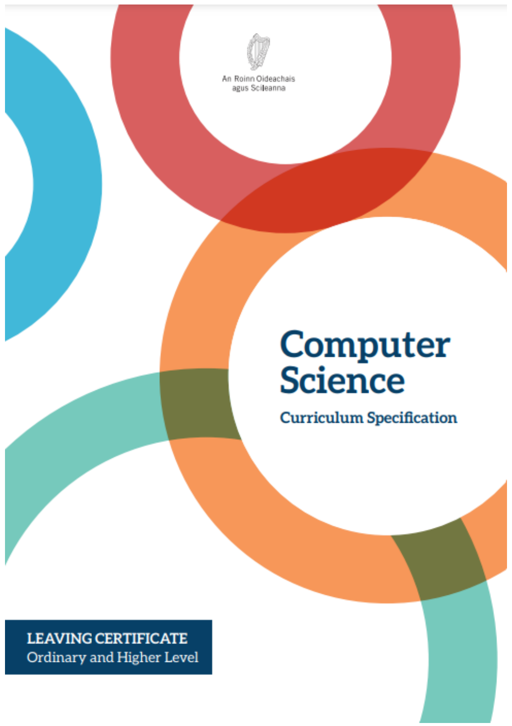
Reports are collected in a digital portfolio along with the computational artefact and must be verified as completed by both the teacher and the student.



Page 11

Create Evaluate Document From the Specification

Students are expected to document, reflect and present on each applied learning task



Page 22

Create

CREATE
implement
the plan

It is not necessary that you finish your project – we are concerned today about understanding the process and the experience

Evaluation and Travel

Oide

An Roinn Oideachais
Department of Education

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers