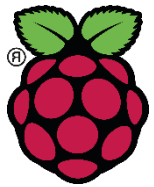
# Perspectives on Teaching Programming for Leaving Certificate Computer Science

Sue Sentance
Raspberry Pi Foundation
28th January 2019
@suesentance

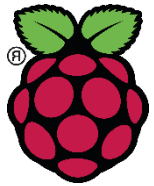


# Structure for this evening

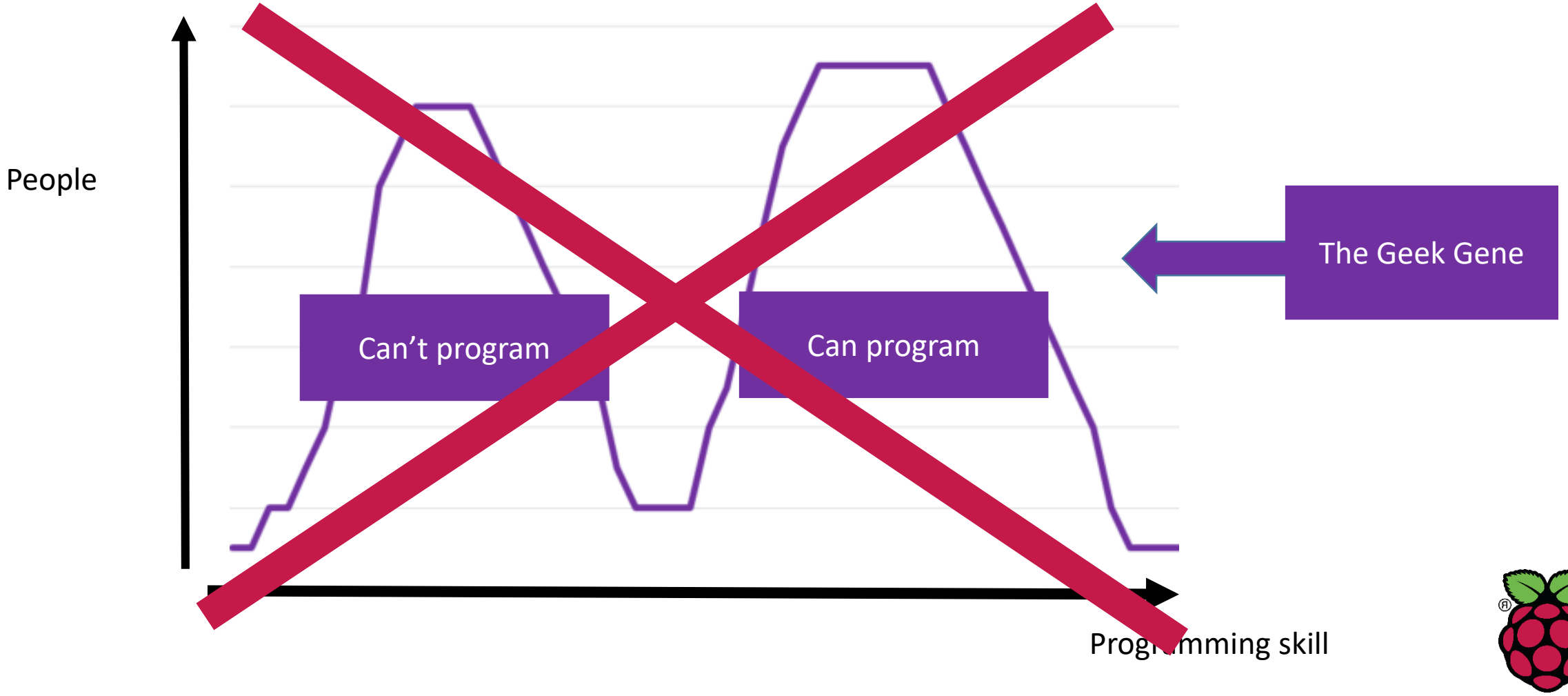
- Can we all learn to program?
- Why programming can be difficult
- Pause for questions

- The myths about programming in school
- Key strategies for a teacher
- Pause for comments and questions

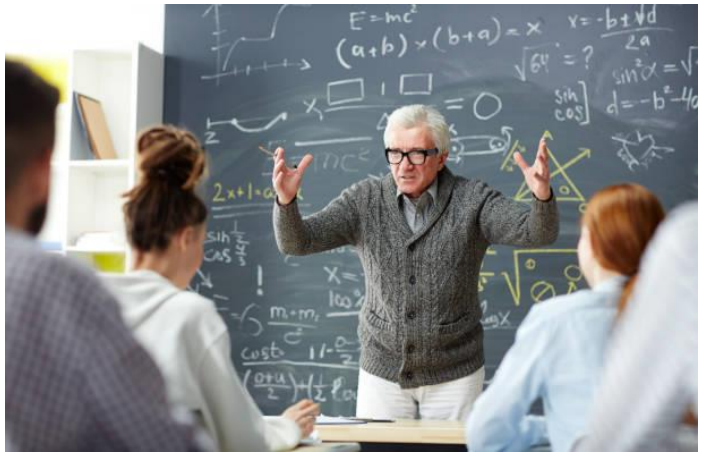
- PRIMM in more detail
- Summary
- Time for questions

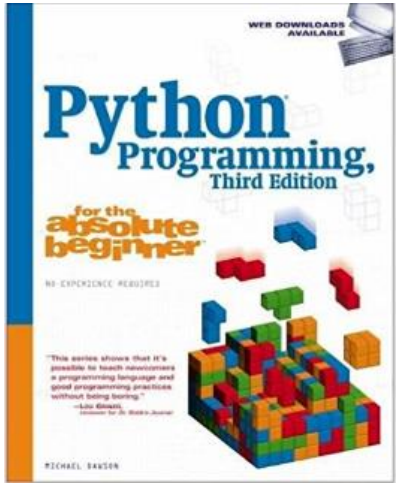


# Do you believe this?

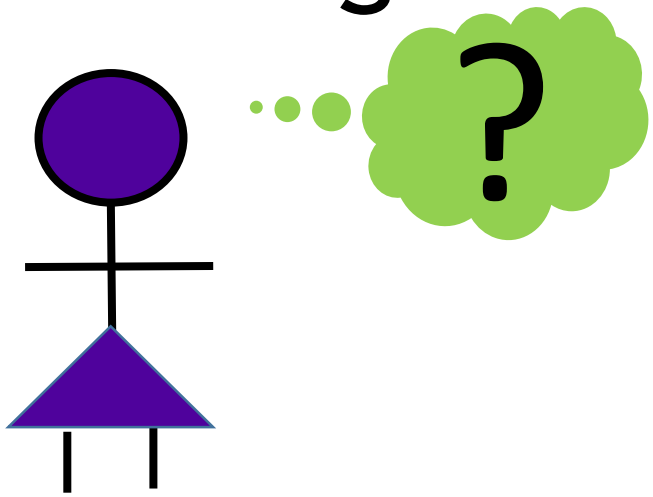


# What’s the best way to learn to program?






# Why programming can be difficult

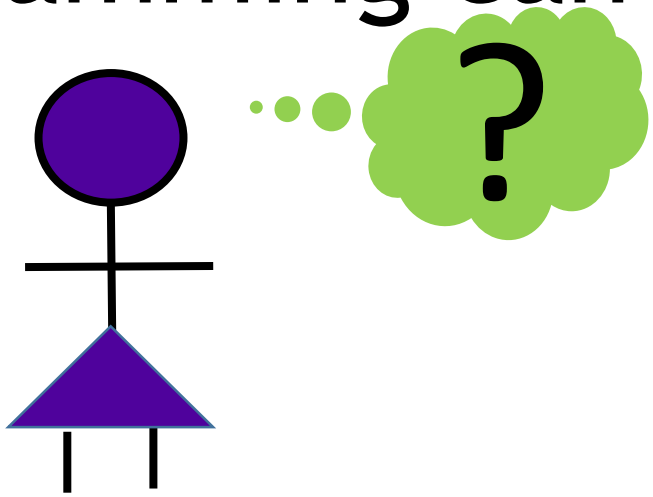



Cognitive Load

1.One line of code (in some languages) can contain lots of concepts to be unpicked and understood. This causes cognitive load

```
myAge = int(input("Please tell me your age .. "))
```

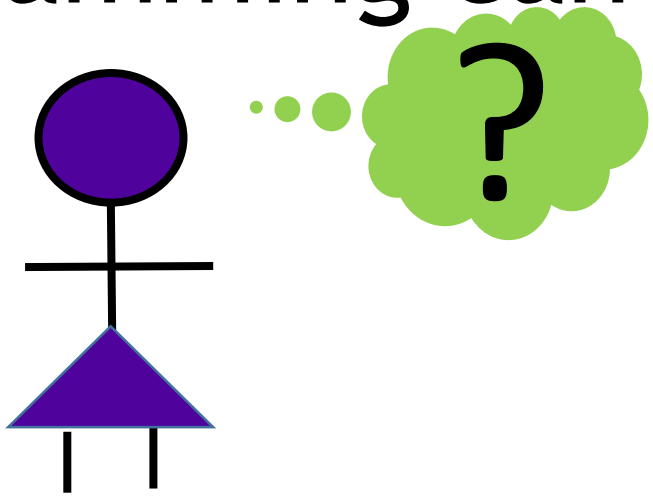
# Why programming can be difficult




Cognitive Load
Writing first?

2. We start writing programs before we are able to read any code and understand (or trace) it. We ask students to copy code when they have no idea how it might work.

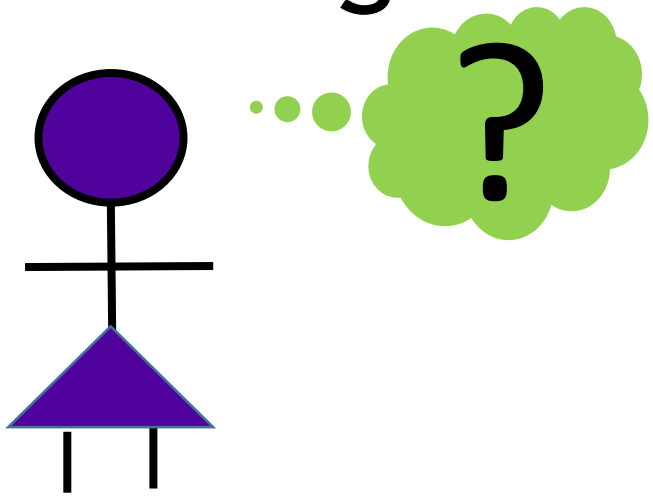
# Why programming can be difficult




Cognitive Load
Writing first?
Resilience

3. Programs don't work first time. To cope with this you need confidence to try again, emotional resilience, and some belief in yourself that it is worth keeping going.

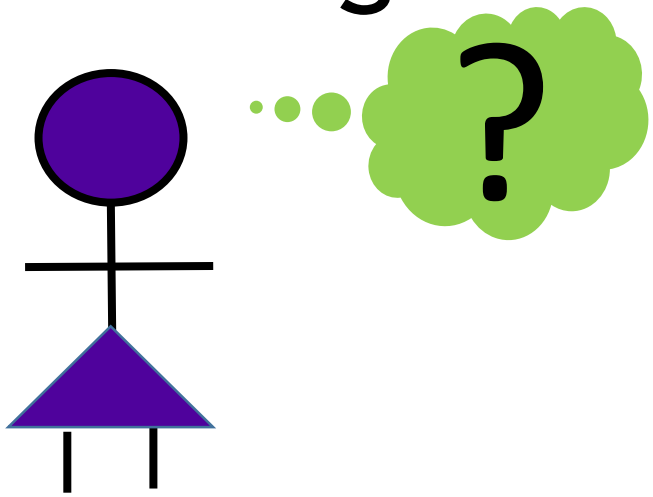
# Why programming can be difficult




Cognitive Load
Writing first?
Resilience
LOA

4.There are (at least) three levels of abstraction when programming - the problem to be solved, how that translates into “code-speak” , and then the “coding”.  Switching between them is not always explicit.

# Why programming can be difficult




Cognitive Load
Writing first?
Resilience
LOA
Mental models

5. You need a mental model of how the computer works when your program executes. If this is not right you can have ALL sorts of problems. Difficult for teachers to detect this. "The computer is magic/is not logical" is one extreme of a mental model.

# What about programming misconceptions?

As with any subject, students can have alternate conceptions that affect their ability to progress.

Some examples in programming*:

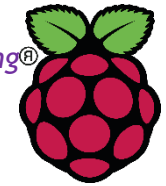
- A variable can store multiple values; it may store the ‘history’ of values assigned to it.
- Both then and else branches are always executed in an if statement
- A while loop’s condition is evaluated constantly. The instant it becomes false, the loop exits.
- Subprograms are executed in the order they are defined in the program text.
- … and others (41 are listed in Sorva’s chapter)

To think about:

a) How you would know a student had that misconception and

b) What would you do to help them to remediate it?

**All these taken from Juha Sorva “Misconceptions and the Beginner Programmer” in Sentance et al (2018). Computer Science Education: Perspectives on Teaching and Learning in School.*

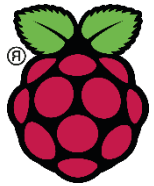


# Pause for comment and questions

We will now have a short break for you to make comments

1. Have you experienced students with these difficulties in school?
2. Are there other key difficulties you have noticed?

3. What is the balance between “understanding” difficulties and “confidence” difficulties in your experience

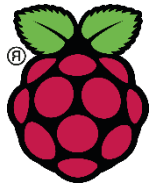


# Some myths

1. Programming is only for geeks, or the child version of geek in school
2. Programming is only for you if you are good at Maths
3. If you build the right tool to support programming it will all be OK

Teachers are key …
Every teacher already has a toolkit of strategies for teaching anything
What particular ones can help with programming?

We need reliable strategies!



# Types of programming learners

## Hobbyists

- Motivated
- Invest own time
- Own choice

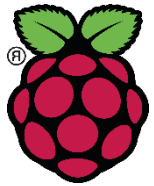
## Employees

- Paid
- Trained by experts
- Have an end goal

## School students

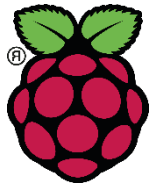
- One lesson a week, for eight weeks of the year
- Large class
- Possibly with a teacher still on a learning curve
- One subject amongst many

Teaching approaches that work for hobbyists and adult programmers may not work for children in school. We need to develop reliable strategies



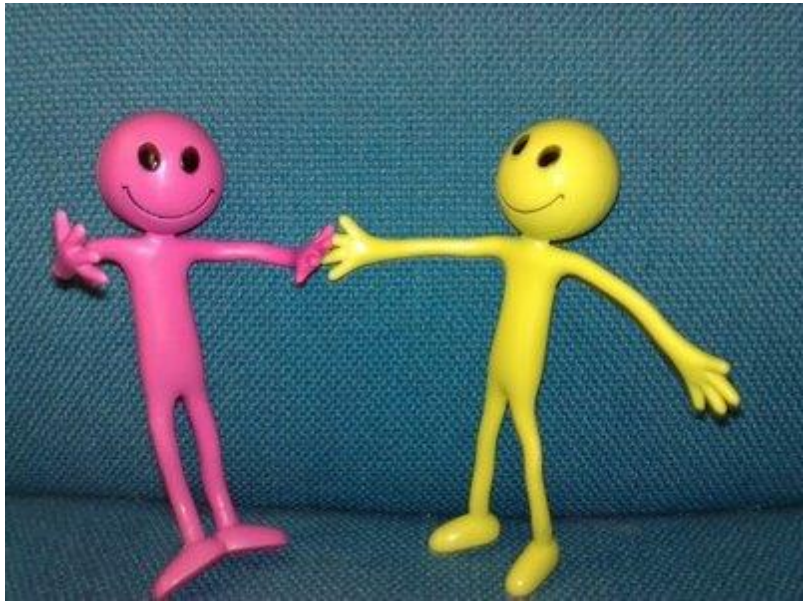
# The teacher’s toolkit

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples
- Parson’s problems
- …. and more ….



# Pair programming

**How does it work?**
One driver, one navigator
Both working on same problem
Change roles every 10-15 minutes
Talk about the problem
Pairing is important

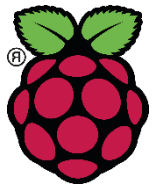


**In the classroom:**
Teacher models good practice
No taking of the mouse or keyboard (without permission)
Point at the screen
Talks through the code

**Evidence**
Shown to work in industry, higher education and increasingly in school
Positive impact on girls

# The teacher’s toolkit

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples (plus with sub-goals)
- Parson’s Problems
- …. and more ….



# Use-modify-create

**Use** – work with an existing program

**Modify** – make changes to it

**Create** – create your own

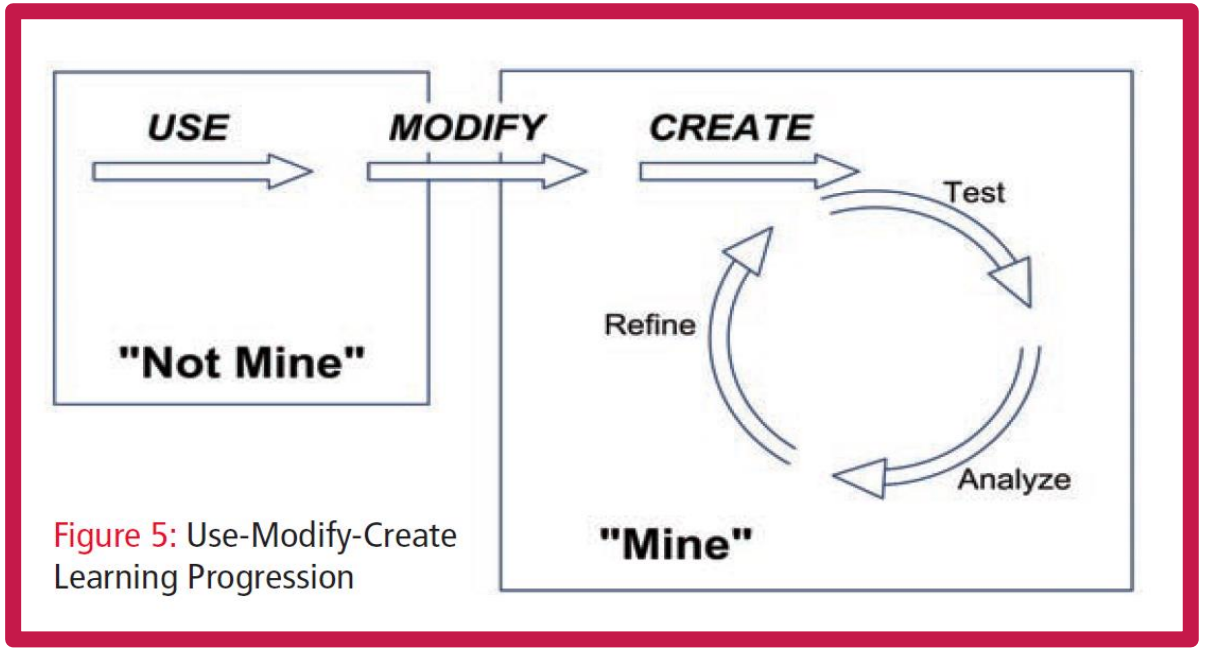


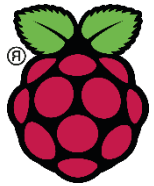

Figure 5: Use-Modify-Create Learning Progression

Great approach for physical computing, for blocks-based programming, and robotics.

Student should be given the program not required to copy code in.

# The teacher’s toolkit

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples (plus with sub-goals)
- Parson’s Problems
- …. and more ….



# The power of “Predict”

Look at this Python (turtle) code

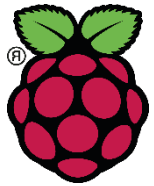
Take a piece of paper

Draw the output

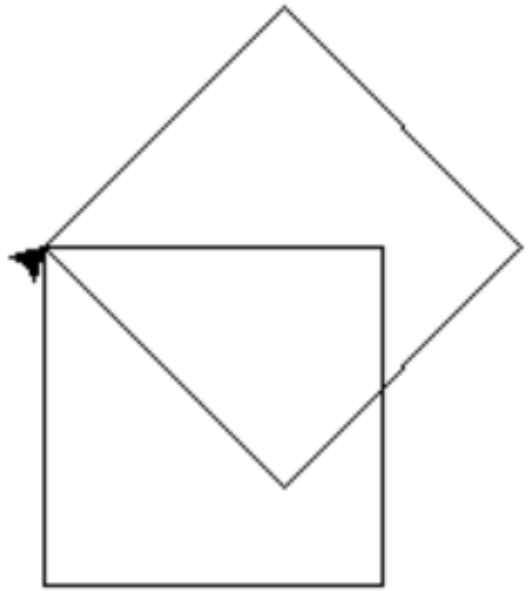
Normally (not in a webinar!) you would discuss your answers with somebody else.

Why might that be useful?

```
from turtle import *

def square():
    for counter in range(4):
        forward(100)
        right(90)

square()
left(45)
square()
```



```
from turtle import *

def square():
    for counter in range(4):
        forward(100)
        right(90)

square()
left(45)
square()
```



Check your answer with the program being executed.

What is different about your answer?

# PRIMM

PRIMM is a way of structuring programming lessons that focuses on:

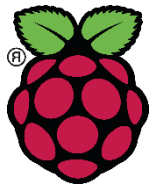
- Reading code before you write code
- Working collaboratively to talk about programs
- Reducing cognitive load by unpacking and understanding what program code is doing
- Using existing starter programs that the learner is not responsible for (if they don’t work!).
- Gradually taking ownership of programs when ready

*“ It was the fact that they were talking and bouncing ideas off each other made it enjoyable and different.”*

# The PRIMM approach

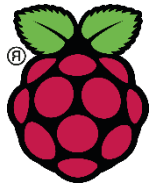
- **Predict** – given a working program, talk about it in pairs or groups. What do you think it will do?
- **Run** – run it and test your prediction
- **Investigate** – get into the nitty gritty. What does each line of code mean? Lots of activities to try here: trace, annotate, explain, talk about, identify parts, etc….
- **Modify** – edit the program to make it do different things
- **Make** – design a new program that uses the same nitty gritty but that solves a new problem

More on PRIMM later



# The teacher’s toolkit

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples (plus with sub-goals)
- Parson’s Problems
- …. and more ….



# Another activity

Look at this code

Choose A, B, C or D

Then using this approach you should then talk in a group about your answer! And then vote again in a group.

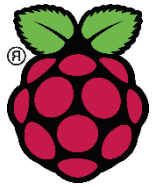
Only then will the teacher discuss the answer

What is the output of the following code?

```
a = 3
b = (a == 3)
print(b)
```

- A. True
- B. False
- C. 3
- D. Syntax error

This is called Peer Instruction



# Peer instruction

Well-evidenced pedagogical strategy
Combination of:

- Flipped learning
- Collaborative working
- Well-chosen MCQs

**Process**

- Step 0: Study topic before session
- Step 1: MCQ question presented to class
- Step 2: Individually decide on answer
- Step 3: All vote
- Step 4: Discuss answer as group
- Step 5: Group decides on answer
- Step 6: All groups vote
- Step 7: Discuss as class

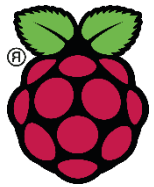
What is the output of the following code?

```
a = 3
b = (a == 3)
print(b)
```

- A. True
- B. False
- C. 3
- D. Syntax error

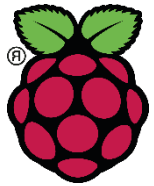
Most effective where there are close distractors and known misconceptions

For more information see
http://peerinstruction4cs.org



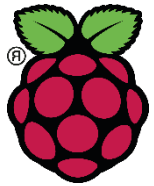
# The teacher’s toolkit

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples (plus with sub-goals)
- Parson’s Problems
- …. and more ….



# Worked examples

- A worked example is a problem where the teacher works through the solution step by step
- Reduces cognitive load – breaking down of information
- A good example of cognitive apprenticeship – experts supporting novices
- Tools exist for worked examples (but teachers are better!)
- A sophisticated version is called “sub-goal labelling” where the individual steps are clearly labelled as goals.



# Worked example - example

Task: Create a program that asks for three numbers, calculates the total and average and displays this information.

**Example dialogue with students**

**Step 1**

- Let's analyse the problem in order to develop a plan…
- Reading the problem again, simply breaking the sentence into sections, it looks as though there may be three steps to the plan …

**Step 2**

- Looking closely at the plan now – how do you calculate an average?
- Well – you total up the numbers first, don't you? And then you can do a division to get the average.
- So the plan needs to be more like this, with four steps, calculating the total BEFORE the average…

**Step 3**

- The first step – ask for three numbers – this is really "read in three numbers". If we do this, where will we store them? In fact do we need to store them?
- No, we could just add each one directly to a running total, and then discard it.
- When reading in a series of numbers, it's always worth asking – do I need to store them all, or can I process each one as it comes in and then throw it away?)
- This allows us to merge the first two steps, so we now have three steps …

1. read in three numbers, adding them to a total
2. calculate average
3. display this information

**Step 4**

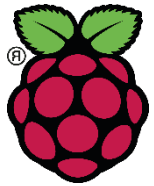
- … etc

(THIS EXAMPLE COURTSEY OF PLAN C / CAS SCOTLAND

https://community.computingatschool.org.uk/resources/5288/single)

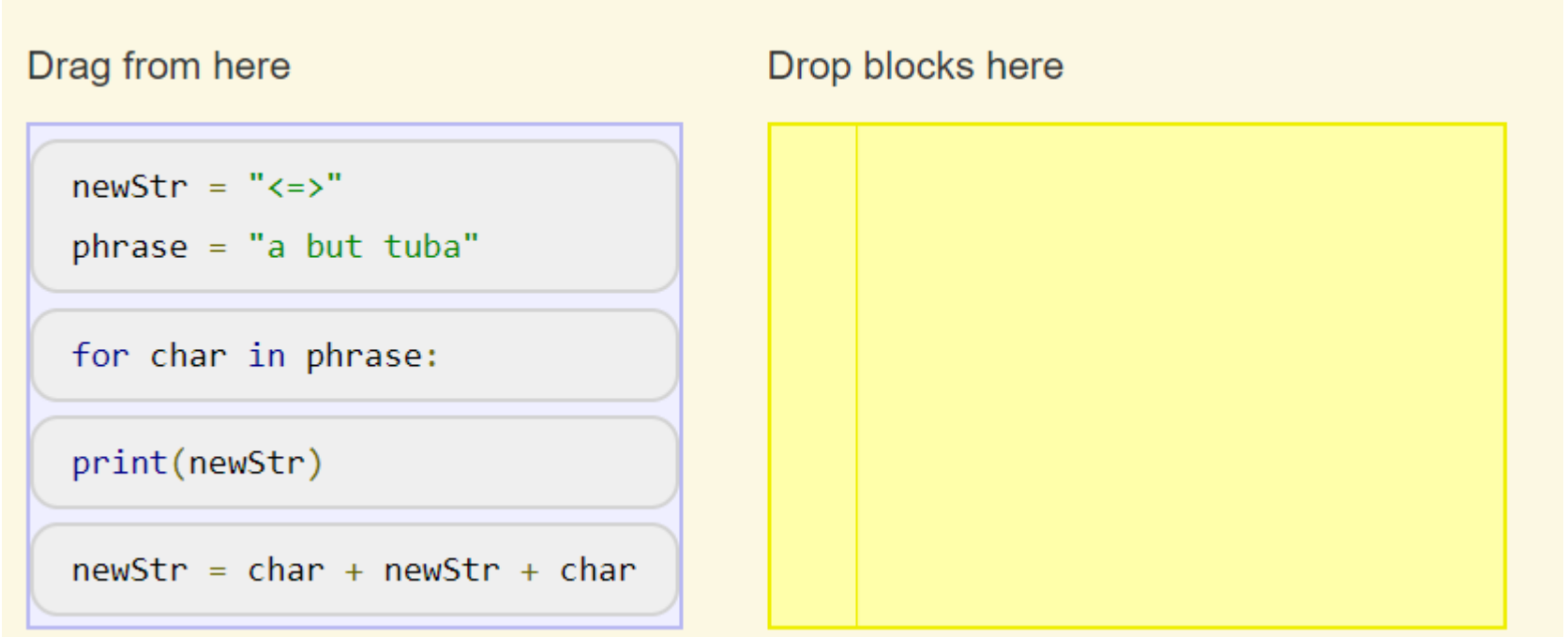
# The teacher’s toolkit

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples (plus with sub-goals)
- Parson’s Problems
- …. and more ….



# Parson's Problems

- Basically mixed up lines of code
- Help students to learn algorithms & can uncover misconceptions
- There are tools for Parson's Problems but bits of paper or the interactive whiteboard is just as good (actually better as you can talk about it)



# In summary

- Pair programming
- Use-modify-create
- PRIMM
- Peer instruction
- Worked examples
- Parson's problems
- …. and more ….

Key principles

- Break down the task
- Use examples
- Work together
- Reading code before writing

Key outcomes

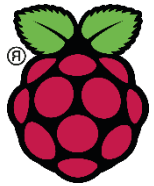
- Cognitive load reduced
- Correct or better mental model
- Confidence in discussing programs
- Better use of programming vocabulary
- Ability to work at different levels of abstraction

# Pause for comment and questions

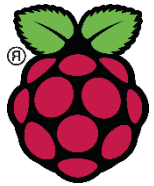
We will now have a short break for you to make comments

Which of these strategies do you think you could put into practice in your teaching?

What other strategies have you tried? Does copying code have a place?



# PRIMM in more detail



# Where did PRIMM come from?

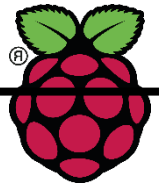
## Areas of research

- Use-Modify-Create
- Tracing
- Levels of Abstraction
- Block Model

## Primary research and experience

Survey in 2014 with 300 teachers around challenges and strategies relating to the teaching of programming – several talked about tracing and debugging activities (Paper is here: https://bit.ly/2Mze2GQ)

Own experience and other teachers around reading code first.



# PRIMM Examples

Predict

Run

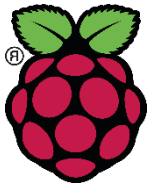
Investigate

Modify

Make

```
def cooking():
    print("Meal planner")
    print()
    print("1. Chicken curry ")
    print("2. Veggie lasagne")
    print("3. Burger and salad")
    print()
    print("Which of these meals is your favourite? (1, 2 or 3) ")
    answer = input()
    if answer == "1":
        print("Chicken curry coming up")
    elif answer == "2":
        print("Veggie lasagne coming up")
    else:
        print("Burger and salad coming up!")
    print("Enjoy!")


cooking()
```

Lesson 3



# PRIMM Examples

Predict

Run

Investigate

Modify

Make

**Task 1: Predict** In pairs, look at the program below and write out what you think might happen when it runs.

```
def starter():
    number = 0
    while number <= 5:
        print("Hello")
        number = number + 1
    print("Goodbye")

starter()
```

What would you expect the computer to do? Write the output exactly as you think it will appear.

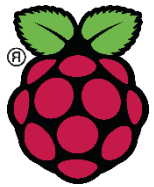
**Task 2**: **Run** Download and run the program and see if it does what you think it might do. You will find it at <insert your shared drive here>

Did the program run as you predicted? _____

What were the differences? _____

Keep starter programs on a shared drive

Students should download, check what they do and compare with prediction



# PRIMM Examples

Predict

Run

Investigate

Modify

Make

Ask different types of questions

**Task 3:** In pairs, work out the answers to the following questions by examining the code and running it a few times.

1. What happens if you don't add any toppings?

.....

2. If you run this program and the user wants to add pineapple, olives and mushrooms, fill in the following table.

| next_topping | toppings |
|---|---|
| | and pineapple |
| olives | |
| mushrooms | |

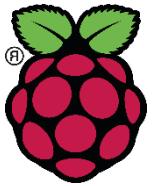
3. Why do you need a while loop?

.....

4. What would happen if you didn't have the line next_topping = ""? (try it and see)

.....

.....



# PRIMM Examples

Predict

Run

Investigate

Modify

Make

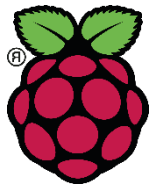
Improve the pizza program so that it doesn't print out "and X" at the end. You will have to add an "if" statement to do this.

Modify the program to have a conversation with somebody about sport. An example is given below but you can add your own questions.

Change your triangle function so that it uses a for loop

Write a function pentagon() to create a pentagon (5 sides) with sides length 200

Write a function hexagon() to create a blue hexagon (6 sides) with sides length 50



# PRIMM Examples

Predict

Run

Investigate

Modify

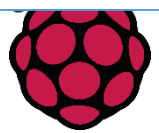
Make

Using your Python skills so far you are going to develop a quiz of 5 questions about Geography

1. First write down 5 questions and 3 possible answers to each
2. Then write your program
3. Test all the different options

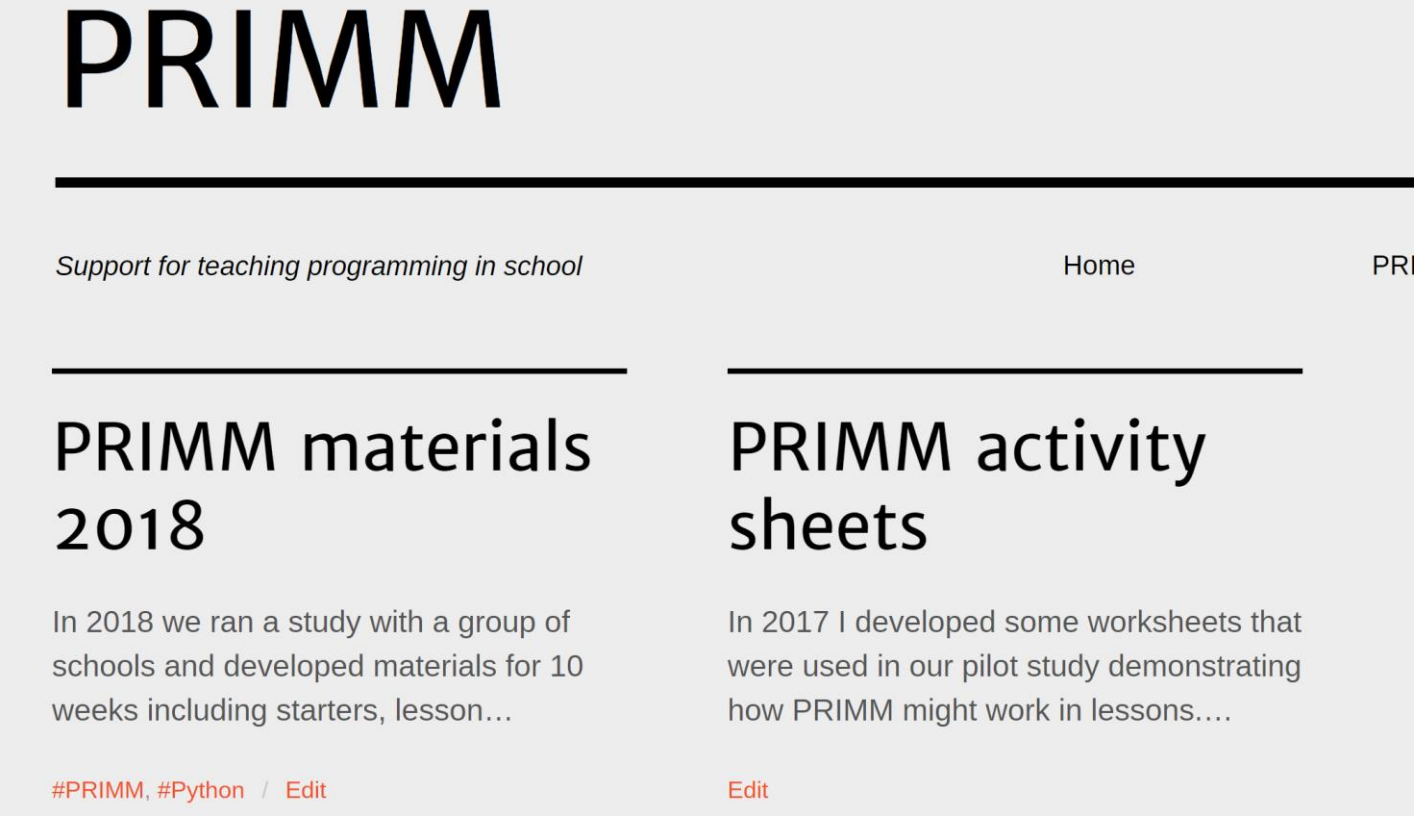
You will be scored on whether it works, whether you have good questions, and how well it is presented (spacing, grammar etc.)

2. Write a version of the game "I went shopping and .. " . Then try it on somebody who can't see the computer to see if they can remember everything the computer has stored from the shopping game? The game should stop when the user presses the enter key rather than adding another item.



# PRIMM Resources

http://primming.wordpress.com




Materials here that were used for the two research studies – and in time we hope to upload teachers' contributions from their own PRIMM resources

# The PRIMM research

## Pilot study 2017

6 teachers

80 students

4-7 lessons

Activity sheets

Teachers edited materials

*“It was amazing! In one lesson, they pretty much all got the concept of a function”*

## Main study 2018

13 teachers

493 students PRIMM

180 in control group

Pre – Post test

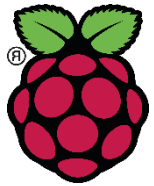
Interviews of teachers

Focus group & journals

# Findings of PRIMM research

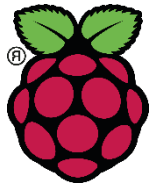
- In the post-test the PRIMM group did significantly better than the control group
- There was no gender difference (not reported)
- Teachers commented on their use of PRIMM:
  - use for differentiation,
  - structure and routine of lessons
  - suitability for low ability students
  - engagement of students
  - learning programming vocabulary

Read more here: https://bit.ly/2sNu0UD - and another paper is under review



# Increasing confidence …

*“. . previously, when they’re writing their own programs, we have so much trouble with syntax errors and half the lesson is just them sorting out syntax. So actually being able to modify it, they can think a little bit more about what their code is doing rather than whether they’ve got a colon in the right place or whatever. And then I think, moving on, once they get to making their own, they’ve got that little bit more confidence that they’ve got a starting point to move on from”*
(Teacher A)



# Differentiation

*“The less able ones [students] enjoyed it (modify) because they got what they were doing when they were at that stage. They were more sure of themselves than they have been in previous Python lessons where they’ve relied on my telling them. This was them doing it themselves . . . the difference was tangible.”* (Teacher I)

# What next?

More PRIMM-style materials around the Block Model

Investigate questions should focus on statements as well as relations between statements to encourage full understanding.

Development of better instruments for testing

## The Block Model (Schulte, 2008)

| | | | |
|---|---|---|---|
| **Macro Structure** | Overall structure of the program text | Understanding the algorithm of the program | Understanding the goal/purpose of the program in context |
| **Relations** | References between separate blocks | Sequence of related actions | Understanding how subgoals are related to overall goals |
| **Blocks** | Regions of interest-blocks like control structures or groups of adjacent statements | Operation of a block | Function of a block, maybe seen as a sub-goal |
| **Atoms** | Language elements | Operation of a statement | Function of a statement. Goal only understandable in context |
| | **Text Surface** | **Program Execution (data and control flow)** | **Functions/goals of the program** |
| | **Structure** | | **Function** |

# Summary

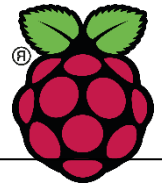
**Overview**

- Can we all learn to program?
- Why programming can be difficult
- The myths about programming in school
- Key strategies for a teacher
- PRIMM in more detail

Comments?
Questions?

## Finally

- England has £80m funding for Computing in school from 2018-2022. Comprehensive resources and training available online (free) as well as face-to-face
- Follow developments at http://teachcomputing.org
- Subscribe to Hello World (http://helloworld.cc ) for useful tips and ideas



# Questions?